ООО «Базальт СПО»
Институт Программных Систем РАН

Объединённая конференция «СПО: от обучения до разработки»

Переславль-Залесский, 19–22 мая 2022 года

Сборник тезисов конференции

Москва,
МАКС Пресс,
2022

УДК 004.91
ББК 32.97
Ч-54

Оформление обложки:
А. С. Осмоловская
Вёрстка: В. Л. Чёрный
Редактура: В. Л. Чёрный

Объединённая конференция «СПО: от обучения до разработки»: материалы конференции / Переславль-Залесский, 19–22 мая 2022 г. / отв. ред. Чёрный В. Л. — М. : МАКС Пресс, 2022. — 208 с.
ISBN 978-5-317-06790-8

В книге собраны тезисы конференции, одобренные Программным комитетом объединённой конференции «СПО: от обучения до разработки».

Издательство ООО «МАКС Пресс»
Лицензия ИД N 00510 от 01.12.99 г.
119992, ГСП-2, Москва, Ленинские горы, МГУ им. М. В. Ломоносова,
2-й учебный корпус, 527 к
Тел. 8(495)939-3890/91. Тел./Факс 8(495)939-3893
Изд. номер № 059

ISBN 978-5-317-06790-8

Программа конференции

19 мая, четверг

Секция: СПО в учебном процессе

11.30-14.00 Заселение, обед, регистрация участников

13:20 Автобус от гостиницы «Переславль»

Дневное заседание
14.00–19.00

14.00 Приветственное слово организаторов

14.20–14.40 Я. Б. Шпунт

Наборы офисных приложений, пригодные для повседневного использования в российской образовательной практике 12

14.40–15.00 Н. Н. Непейвода

На пути к несписываемым задачам: сравнение математических пакетов 16

15.00–15.20 М. В. Маркушевич, А. Н. Краснов

Несостоятельность принципа инвариантности методики обучения информатике в основной школе относительно программного и аппаратного обеспечения 17

15.20–15.40 Е. А. Синельников

Основные проблемы вовлечения студентов в Open Source .. 21

20 мая, пятница

Секция: СПО в учебном процессе

09:30 Автобус от гостиницы «Переславль»

Утреннее заседание
10.00–13.00

13:00 Автобус в гостиницу «Переславль»

13.00–15.00 Перерыв на обед

14:40 Автобус от гостиницы «Переславль»

Вечернее заседание
15.00–19.20

15.00–15.20 А. Н. Непейвода

15.20–15.40 В. Л. Симонов, Е. А. Лапшина

15.40–16.00 Д. А. Киселёв, В. Л. Симонов

16.00–16.20 В. Л. Симонов, Е. А. Лапшина и др.

16.20–16.40 А. А. Маркина

16.40–17.00 И. В. Воронин, А. Дарсавелидзе

17.00–17.20 Кофе-пауза

17.20–17.40 И. Н. Зыкин

21 мая, суббота

Секция: Разработка свободного ПО

09.30-10.00 Регистрация участников

09:30 Автобус от гостиницы «Переславль»

Утреннее заседание
10.00–13.00

10.00–10.20 А. В. Федорчук

Linux для пращуров и внуков 133

10.20–10.40 А. А. Якушин

Протестантская этика Макса Вебера и дух свободного программного обеспечения 135

10.40–11.00 А. А. Панюкова, П. Д. Арлинский, Д. Петров

Студенческий проект: программные продукты на основе платформы ShariX 137

11.00–11.20 Н. В. Шмырёв

Перспективы развития открытых проектов в области машинного обучения 140

11.20–11.40 Кофе-пауза

11.40–12.00 Д. А. Маракасов

Repology: мониторинг пакетных репозиториев 142

12.00–12.20 А. В. Бондарев

Интернет вещей на базе СПО 144

12.20–12.40 Р. Р. Шаниязов

Эффективное взаимодействие: протокол `GPRS` и язык `protobuf` 147

12.40–13.00 И. Е. Панченко

Куда идут слоны 152

13:00 Автобус в гостиницу «Переславль»

13.00–15.00 Перерыв на обед

14:40 Автобус от гостиницы «Переславль»

Вечернее заседание
15.00–18.40

22 мая, воскресенье

Секция: Разработка свободного ПО

09:30 Автобус от гостиницы «Переславль»

Утреннее заседание
10.00–13.00

10.00–10.20 Е. К. Шестепёров

Автоматизация процессов в рамках тестирования сборочных заданий для стабильных репозиториев ОС «Альт» 182

10.20–10.40 Черноног М. А.

Автоматическое тестирование дистрибутивов ОС «Альт» с использованием OpenQA 185

10.40–11.00 Р. Г. Ставцев

Проблемы совместимости при построении аппаратных платформ на примере SoC Baikal-M 186

11.00–11.20 Р. Г. Ставцев

Перенос ROS с Ubuntu на решения «Базальт СПО» 187

11.20–11.40 Кофе-пауза

11.40–12.00 Е. А. Синельников

Сценарии миграции доменных инфраструктур 187

12.00–12.20 В. А. Синельников

Текущий статус и перспектива развития «Групповых политик» в ОС «Альт» 191

12.20–12.40 И. В. Савин

Группы в группах 196

12.40–13.00 В. Н. Благовещенская

Тестирование инструментов управления групповыми политиками Active Directory и инструментов администрирования, совместимых с Microsoft Active Directory в ОС «Альт» в рамках гранта РФРИТ 200

14:00 Автобус в Москву

Вне программы

Секция: СПО в учебном процессе

Яков Шпунт
Москва, Comnews Group

Наборы офисных приложений, пригодные для повседневного использования в российской образовательной практике. Сравнение функциональных возможностей наиболее распространённых наборов офисных приложений для Linux

Аннотация

OpenOffice.org и LibreOffice наиболее известны среди свободных офисных приложений. Но они далеко не единственные, которые существуют для Linux. Автор предпринял попытку выявить сильные и слабые стороны офисных приложений для Linux и очертить возможные сферы применения.

Для Linux существует несколько наборов офисных приложений, пригодных для реального использования в образовательных учреждениях. Тем не менее, у каждого из них есть свои особенности и слабые места, что оказывает влияние на сферу применения.

Кого пришлось исключить

Несколько наборов были исключены из рассмотрения в силу разных причин. MS Office Online, Gapps и Zoho представляют собой облачные сервисы, работа с которыми без подключения к Интернету невозможна. Учитывая то обстоятельство, что подключиться к Сети можно далеко не везде, данный недостаток был сочтён существенным.

WPS Office был снят с рассмотрения вследствие наличия вредоносного ПО в бесплатной версии. Calligra Suite же просто непригодна для работы вследствие очень плохой совместимости с другими офисными приложениями.

LibreOffice и OpenOffice.org

Оба набора по факту являются форками одного проекта. Однако, LibreOffice развивается несколько более динамично и более функцио-

нален (в частности, намного лучше реализована поддержка макросов на VBA из MS Office), а также поддерживает разные виды пользовательского интерфейса (традиционный и ленточный). Однако в целом состав и функциональные возможности близки. Существует для разных программных и аппаратных платформ (включая российские).

Включает целый комплекс расширенных функций, включая редактор формул и модуль работы с индексами. Есть СУБД и довольно мощный редактор векторной графики. В целом данные продукты являются универсальными офисными редакторами, пригодными для любых задач. Есть шаблоны, позволяющие создавать документы в соответствие со стандартом ЕСКД.

Главный недостаток — проблемы при импорте документов в формате приложений MS Office. Особенно критичны они в Impress, который очень существенно искажает внешний вид слайдов PowerPoint.

Данный продукт целесообразно использовать студентам старших курсов, магистрантам, аспирантам и профессорско-преподавательскому составу. Для школьников и учащихся ССУЗ возможности данных приложений избыточны.

GNOME Office

По факту данный набор является разрозненными программами, которые базируются на общем тулките GTK. Официальные версии приложений есть только для Linux, хотя раньше поддерживался весьма широкий спектр платформ, включая мобильные. Поддерживает российские аппаратные платформы.

Главное достоинство ключевых приложений, за исключением Evolution — низкие системные требования. AbiWord вполне работоспособен даже на компьютере со 128 Мбайт оперативной памяти. При этом функциональные возможности включают практически все необходимые функции. Возможно использование подключаемых модулей, применение которых расширяет штатную функциональность. Возможности Gnumeric и вовсе практически те же, что и у Calc или Excel.

Недостатки набора связаны с издержками слабой координации разработчиков. В итоге если Gnumeric или Evolution вполне функциональные и зрелые приложения, то AbiWord им серьёзно уступает и пригоден лишь для начального обучения навыкам работы с текстом. Нет редактора презентаций.

OnlyOffice/Р7 Офис

Продукт от российской компании «Новые коммуникационные технологии», с 2016 года распространяется по лицензии GNU GPL 3. Есть улучшенный вариант Р7, входящий в Реестр российского ПО. Также Р7 поддерживает стандарт ЕСКД «из коробки». Но при этом не поддерживаются отечественные аппаратные платформы, за что разработчиков даже раскритиковало Минцифры.

Изначально данный продукт создавался для нужд небольших компаний. При этом за образец были взяты Gapps. Продукт изначально был предназначен для развёртывания онлайн, в том числе в корпоративной сети, и функция редакторов документов появилась не сразу.

Функциональность приложений достаточная, но при этом «лишних» функций, которые будут заведомо не востребованными, там нет. Но стоит иметь в виду, что не включён целый ряд возможностей, которые необходимы в образовательной и научной практике. Впрочем, продукт довольно активно развивается, и функциональность растёт.

Главное достоинство — полноценная и при этом корректная поддержка как ODF, так и OpenXML форматов. Практически идеальное решение для тех, у кого большая библиотека унаследованных документов или кому необходимо обмениваться ими с внешним миром.

Недостаток — довольно высокие системные требования. Впрочем, работа с десктопным редактором будет вполне приемлемой даже на системе с Intel Atom 28xx, имеющей 2 Гбайта оперативной памяти. Облачная версия работает в любом более-менее современном браузере.

Вполне самодостаточный продукт, хорошо совместимый с MS Office. Вполне пригоден для всех звеньев образовательной системы, от младшей школы до вузов. Не вполне пригоден разве что для подготовки диссертаций и монографий, где требуется составление предметных и именных указателей.

Мой Офис

Бесплатен для образовательных учреждений. Также именно его наиболее часто выбирают для замены приложений Microsoft при реализации программ импортозамещения в госучреждениях и госкомпаниях. Входит в Реестр российского ПО. Поддерживает российские аппаратные платформы.

Текстовый и табличный процессоры имеют достаточную для работы с офисной документацией функциональность. Последняя версия поддерживает стандарт ЕСКД «из коробки». Возможно развёртывание в локальной сети или частном облаке. Есть версии для Windows, Linux, Android, iOS, «Аврора», mac OS. Очень хорошее качество документации. В состав дистрибутива входит подробное руководство, есть учебник, ориентированный на младших школьников.

Полноценное средство для редактирования презентаций пока находится в стадии разработки. Есть жалобы на проблемы с искажением оформления документов при импорте.

Вполне самодостаточный продукт, неплохо совместимый с MS Office. Пригоден для всех звеньев образовательной системы.

Softmaker FreeOffice

Данный набор полностью бесплатен для всех категорий пользователей. Включается в состав некоторых дистрибутивов РОСА.

Главное достоинство — компактность. Версия для Linux или Windows занимает на диске меньше, чем объём дистрибутива LibreOffice. Существует 4 вида сменных интерфейсов. Имеется официальная версия для Android, в том числе рассчитанная на экраны высокого и низкого разрешения.

Хорошая функциональность, уступает только Open/LibreOffice и превосходит все остальные. Например, текстовый процессор поддерживает работу с указателем. «Из коробки» стандарт ЕСКД не поддерживает, но возможно использование шаблона для LibreOffice.

Недостатки — невысокое качество системы проверки правописания, находили ошибки в словарях; не поддерживает российские аппаратные платформы.

Литература

[1] Семён Лыткин. Тем, кто устал от готики. https://www.dgl.ru/reviews/tem-kto-ustal-ot-gotiki_20111.html

[2] Власти пристыдили «1С», «Р7-офис» и «Парус» за медленную миграцию на российские чипы и посулили денег. https://www.dgl.ru/reviews/tem-kto-ustal-ot-gotiki_20111.html,https://www.cnews.ru/news/top/2021-08-17_vlasti_prosyat_uskorit_perenos

Непейвода Н. Н
Переславль-Залесский, ИПС РАН

На пути к несписываемым задачам: сравнение математических пакетов

Аннотация

Одним из симптомов глубокого кризиса высшего инженерного образования в России, превратившегося за последние годы в катастрофический обвал и жалобы студентов: нас не учат, а разучивают, мы на выходе умеем и знаем меньше, чем при поступлении, — является тотальное списывание. Студенты имеют базы решений задач, курсовых и дипломных работ, не обновлявшихся по 10–50 лет (впрочем, для попадания в студенческие базы шпаргалок достаточно не обновлять два года), а преподаватели легально списывают свои задачи и темы из материалов предыдущих лет.

Одним из решений является здесь резкое повышение сложности заданий с использованием математических программ при официальном разрешении студентам пользоваться этими программами для их решения. Умение вручную брать интегралы становится всё менее востребованным.

Например, в задаче

$$\int \frac{3\sin\alpha + 3\cos 2\alpha - 2\sin^3\alpha}{8\cos^3\alpha - 6\cos\alpha}\, d\alpha$$

обычными средствами поиска и копипаста невозможно распознать практически табличный интеграл, а тупое применение математической программы даст настолько неуклюжий ответ, что сразу станет ясно: студент не приложил свой ум. Подобные задачи, внешне абсолютно различные, легко даже вручную создать индивидуально для каждого студента перед каждой контрольной. А уж если у преподавателя есть маленький опыт в программировании, он их будет порождать полуавтоматически (полностью автоматизировать не стоит, быстро студенты вновь поймают). Тем самым одновременно рещаются две проблемы: индивидуализация рутинных заданий и быстрый контроль качества решений задач.

Качественные результаты сравнения ведущих математических программ можно резюмировать следующим образом.

Maxima (свободная). Лёгкое обучение, достаточна по возможностям для вузов, где математика ограничивается математическим анализом и дифурами средней сложности.

Maple (проприетарная). Лёгкое обучение и пользование, достаточно широкие возможности (чуть шире, чем у Maxima), возможности оформления решений значительно выше.

Mathcad (проприетарная). Достаточно лёгкая в обучении и удобная в использовании, подходит для инженерных вузов с повышенной численной математикой.

Mathematica (проприетарная, дорогая). Лёгкая в использовании, более трудная в обучении, чем Maple. Оформление результатов уступает Mathcad, но на высоком уровне. Имеет нечисленные пакеты.

SageMath (свободная). Трудна в освоении, не очень удобна в использовании, но исключительно богата в области тонких формальных и численных методов и особенно нечисленной математики. Свои возможности оформления достаточно скромные по сравнению с Mathematica, но имеется возможность удобного экспорта результатов в TeX. Подходит для вузов с повышенным курсом математики, включающим нечисленную (алгебру, логику, алгебраическую геометрию).

Любой из этих пакетов может быть использован для создания новых задач и тем работ, но под санкции не могут подпасть лишь первый и последний.

Михаил Маркушевич, Алексей Краснов
Город Москва, Государственное бюджетное общеобразовательное учреждение города Москвы «Школа № 1352»

Несостоятельность принципа инвариантности методики обучения информатики в основной школе относительно программного и аппаратного обеспечения

Аннотация

В настоящей работе авторами рассматривается вопрос состоятельности (несостоятельности) принципа инвариантности методики обучения информатике в основной школе относительно типа программного

обеспечения, используемого в учебно-воспитательном процессе, а также относительно типа аппаратного обеспечения (мобильные устройства, настольные компьютеры различных конфигураций). До недавнего времени принцип инвариантности методики обучения в педагогическом сообществе считался не обсуждаемым и очевидным, однако к настоящему моменту накопилось достаточно причин подвергнуть его более подробному анализу на состоятельность. Далее авторы рассматривают, какой тип программного обеспечения является оптимальным для поддержки учебного процесса по информатике в основной школе при обучении различным информационным технологиям (создания и обработки текстовой информации, растровой и векторной графической информации, звуковой и видеоинформации и т. д.) и делают выводы об очевидных преимуществах выбора отечественного свободного программного обеспечения в качестве базового.

В современной российской педагогике достаточно прочно сформировался принцип инвариантности методики обучения информатике относительно программного и аппаратного обеспечения, используемого в учебном процессе. Интересно отметить, что даже авторы некоторых современных научных статей [4] не подвергают его сомнению и не пытаются смотреть на данную проблему под другим углом зрения.

В чём причина такой удивительной стойкости рассматриваемого нами принципа? Для того, чтобы ответить на этот вопрос, необходимо мысленно вернуться в то время, когда складывалась отечественная школьная информатика и формировались все базовые принципы, на которых она строилась. Речь идёт о конце 80-х и 90-х годах прошлого века, которые, в том числе, можно характеризовать периодом правового хаоса, в условиях которого мало кто всерьёз относился к вопросам соблюдения авторского права в области программного обеспечения. Кроме того, в то время проект по разработке свободного программного обеспечения GNU только появился и начал своё развитие (1983 год) [1], в нём участвовали только профессиональные программисты. Только в 1991 во время обучения в Хельсинкском университете Линус Торвальдс заинтересовался операционными системами и занялся написанием кода для ядра свободной операционной системы Linux [2], в то время как, уже в 1990 году Microsoft Corporation выпустила проприетарную операционную систему Windows 3.0 для персональных компьютеров [3] и Microsoft Office 1.0, которые могли использовать обычные пользователи.

Таким образом, период становления отечественной школьной информатики совпал с периодом тотального доминирования проприетарного программного обеспечения и одновременно правового нигилизма в области авторского права в Российской Федерации.

Впервые российское педагогическое сообщество столкнулось с проблемой соблюдения авторского права, а также, как следствие, с проблемой выбора типа программного обеспечения для поддержки учебного процесса в общеобразовательной школе по информатике в 2007 году, когда бывший директор Сепычевской средней школы Пермского края А.М. Поносов был приговорён судом к штрафу за использование нелицензионных копий ОС Windows и Microsoft Office на школьных компьютерах [5]. Для предотвращения повторения подобных ситуаций в 2008 году в российские школы был направлен для тестирования и использования как пакет свободного программного обеспечения (ПСПО) на базе отечественной операционной системы Alt Linux, так и пакет «Первая помощь», содержащий в себе очень серьёзный стек проприетарного программного обеспечения ведущих иностранных разработчиков (Microsoft Corporation, Adobe Inc., Macromedia Inc. и т.п.).

К сожалению, российское педагогическое сообщество, в большей своей части не было готово на тот момент к квалифицированному и эффективному применению свободного программного обеспечения в учебном процессе в виду отсутствия доступных курсов повышения квалификации по данной тематике, а также относительно малому количеству учебно-методической литературы, посвящённой методике обучения школьников информатике на основе использования свободного программного обеспечения. Более подробно анализ основных затруднений, возникающий при переводе учебного процесса на СПО описан авторами в работе [6]. Таким образом, в тот момент также не возникло причин к пересмотру принципа инвариантности методики обучения относительно программного обеспечения в связи практически тотальным ведением преподавания информатики в отечественных школах на базе дорогостоящего импортного проприетарного программного обеспечения и после 2008 года.

Только в период после 2014 года, с момента введения экономических санкций рядом западных государств против, в том числе, российских вузов [7], свободное программное обеспечение стало постепенно занимать соответствующее место как в российских школьных учебниках по информатике и информационным технологиям, так и использоваться всё большим числом преподавателей для поддержки

учебного процесса, что поставило перед педагогическим сообществом задачу вернуться к рассмотрению вопроса состоятельности принципа инвариантности.

Отметим, что характеристиками программного обеспечения, влияющими на методику обучения, с точки зрения авторов, являются следующие:

- Идеология разработки и последующей технической поддержки ПО (свободное или проприетарное);
- Минимальные системные требования к hardware персональных компьютеров, на которых предполагается использование ПО (определяет доступный парк оборудования и, следовательно, список образовательных организаций, в которых возможно использовать данную методику обучения);
- Кроссплатформенность или ориентация только на один тип ОС (определяет доступный парк оборудования и, следовательно, список образовательных организаций, в которых возможно использовать данную методику обучения);
- Тип персональных компьютеров (планшетные устройства, настольные ПК), на которых предполагается использование ПО;
- Простота, интуитивная понятность или загруженность интерфейса пользователя (влияет на выбор для определённого уровня образования);
- Функционал данного ПО, определяющий деление ПО на профессиональное или «любительское» (дифференциация между специальным и общим образованием);
- Наличие или отсутствие хорошей методической поддержки в плане его использования в учебном процессе (наличие учебно-методической литературы, Интернет-источников);
- Является онлайн-сервисом или офлайн программным обеспечением, не зависящим от подключения к сети Интернет;

Таким образом, с точки зрения авторов, можно говорить о том, что принцип инвариантности методики обучения информатике в общеобразовательной школе относительно используемого в учебном процессе программного и аппаратного обеспечения является несостоятельным по причине наличия большого количества факторов зависимости методики обучения информатики от типа используемого программного и аппаратного обеспечения.

Литература

[1] Проект GNU https://ru.wikipedia.org/wiki/P§CЃP«PхPəCЂ_GNU;

[2] Linux https://ru.wikipedia.org/wiki/Linux;

[3] Windows 3.x https://ru.wikipedia.org/wiki/Windows_3.x;

[4] Королёва Н. Ю. Формирование умений работы с цифровым звуком в школьном курсе информатики // Информатика в школе. 2021. № 10. с. 28–31;

[5] Поносов Александр Михайлович https://ru.wikipedia.org/wiki/Поносов,_Александр_Михайлович;

[6] Fedosov A., Markushevich M., Gubina T. Free Software at School and the Educational Process: Current State, Systematization of Typical Problems. In: Silhavy R., Silhavy P., Prokopova Z. (eds) Software Engineering Application in Informatics. CoMeSySo 2021. Lecture Notes in Networks and Systems, vol 232. Springer, Cham. https://doi.org/10.1007/978-3-030-90318-3_19

[7] МГТУ Баумана остаётся без Windows: Microsoft отказалась поставлять ему своё ПО // https://www.cnews.ru/news/top/2020-12-09_mgtu_baumana_ostalsya_bez

Евгений Синельников
Саратов, ООО «Базальт СПО»
Проект: ALT for students https://www.altlinux.org/Students

Основные проблемы вовлечения студентов в Open Source

Аннотация

Доклад посвящён позитивным и негативным сценариям вовлечения студентов в разработку проектов с открытым исходным кодом. В докладе рассмотрены возможности и препятствия к использованию открытых технологий, как основы для обучения будущих разработчиков. Представлены особенности и ограничения подходов к разработке с привлечением студентов. Изложены варианты развития данного образовательного направления.

«Открытые технологии» в контексте рассматриваемого вопроса вовлечения студентов в Open Source разработку — это технологии, распространяемые по свободным лицензиям. Целевой аудиторией в рассматриваемом вопросе являются студенты большинства российских вузов (не только классических и не только столичных). Основная задача такого вовлечения состоит в подготовке будущих кадров для разработки системного и прикладного программного обеспечения под свободные операционные системы. Прежде всего, под различные Linux-дистрибутивы и для различных аппаратных платформ.

Практика преподавания информационных технологий для различных дисциплин инженерных специальностей (прикладное и системное программирование, архитектуры операционных систем и др.), анализ реализации студентами практических задач и обратная связь от студентов дают основание рассмотреть возможности и препятствия к использованию открытых технологий в современном учебном процессе.

В сложившемся подходе к преподаванию программирования основными, наиболее значимыми препятствиями и главными ограничительными противоречиями вовлечения студентов в развитие открытых технологий, являются следующие ключевые моменты:

- Отсутствие практики коллективной работы в рамках выполнения основных практических заданий, включая отсутствие знакомства с соответствующим инструментарием, прежде всего, с системой контроля версий git;

- Ориентация типовых учебных заданий на повышение навыков написания простых, однофайловых проектов без освоения различных систем сборки (make, cmake, autotools и т. д.), а также минимального оформления архитектуры, разрабатываемых проектов;

- Отсутствие нацеленности типовых учебных заданий на повышение навыков чтения исходного кода чужих программных проектов различной степени сложности.

Среди технических ограничений, в данном случае, стоит также отметить отсутствие стандартных, полностью согласованных с соответствующими системами сборки, сред разработки, позволяющих понизить порог вхождения, а среди не технических — отсутствие актуализированной русскоязычной технической документации.

Возможности для развития и требования к подготовке

Одной из важнейших возможностей для развития студентов, как будущих разработчиков, при их вовлечении в открытые технологии, заключается в доступности к изучению оригинальных программных и технических решений в том виде, в котором эти технологии используются на практике. Ряд проприетарных компаний-разработчиков также предоставляют такие возможности (например, Microsoft или Oracle) только в ограниченном, хотя и достаточном для изучения объёме.

При этом перед студентом возникает ряд неявных требований к его подготовке:

- письменный английский язык — большая часть актуальной документации, комментарии в коде, включая имена переменных, функций, классов и т. д., комментарии к зафиксированным изменениям в системах контроля версий, публичное общение осуществляются, в основном, на английском;
- готовность и умение читать и вникать в чужой исходный код;
- иметь доступ или возможность развернуть у себя в домашних условиях учебную инфраструктуру, включая комплект подобранных инструментов для обучения: конкретный вариант дистрибутива плюс конкретные средства разработки.

Конкретные детали в тех или иных ограничениях зависят от решаемой задачи и стека технологий, выбранных для её решения. Таким образом, можно сказать, что возможности для развития студента в области открытых технологий имеются, но порог вхождения оказывается сильно завышен, по отношению к проприетарным технологиям. В этом есть и свои плюсы, поскольку такие условия обеспечивают автоматический отбор наиболее выдающихся кадров. С другой стороны, это означает, что критическая масса необходимого числа разработчиков в сфере открытых технологий не накапливается. И последнее, в свою очередь, приводит к накапливанию проблем в инструментарии и документировании.

Дальнейшее развитие разработки в области открытых технологий и свободного программного обеспечения, в частности, следует начинать с преодоления вышеозначенных ограничений. Преодоления за счёт снижения порога вхождения не только на техническом, но и на организационном уровне.

Для разработки под операционные системы семейства «Альт» соответствующие ограничения также имеют определённую специфику, связанную с тем, что большая часть сообщества ALT Linux Team преимущественно общается и взаимодействует на русском языке. Поэтому встроенный инструментарий для разработки и документирование средств разработки необходимо развивать согласованно на двух основных языках — русском и английском. Кроме того, специфичный стек средств разработки для сборки пакетов, например «из git в rpm с помощью gear» требует детальной методической проработки.

Кушниренко А. Г.[(1)], Леонов А. Г.[(1–4)], Мащенко К. А.[(1–2)], Орловский А. Е.[(1–2)], Райко М. В.[(1)], Шляхов А. В.[(1–2)]
г. Москва, Федеральное государственное учреждение «Федеральный научный центр научно-исследовательский институт системных исследований Российской академии наук» (ФГУ ФНЦ НИИСИ РАН) (1)
Московский педагогический государственный университет (2)
Государственный университет управления (3)
Федеральное государственное бюджетное образовательное учреждение высшего образования «Московский государственный университет имени М.В.Ломоносова» (4)

Проект: КуМир, Мирера `https://mirera.ru/`

Опыт интеграции цифровой образовательной среды КуМир в платформу Мирера

Аннотация

Пандемия продемонстрировала необходимость создания и расширения дистанционных образовательных решений с доступом через интернет. В статье сформулирована необходимость поддержки школьного алгоритмического языка (цифровой образовательной среды КуМир) в качестве web-приложения. Рассмотрены вопросы интеграции компилятора КуМира в цифровую образовательную платформу (ЦОП) Мирера, изложен опыт преподавания курсов по информатике и программированию на школьном алгоритмическом языке с использованием ЦОП Мирера. Предложены пути дальнейшей интеграции КуМир и ЦОП Мирера с поддержкой единой системы аутентификации, автоматизированной проверкой выполненных заданий в ЦОП Мирера и визуализацией графических результатов.

Введение

В сентябре 1985 года в 9–10-х классах школ СССР появился новый предмет «Основы информатики и вычислительной техники». Изначально предполагалось преподавать в так называемом «безмашинном» варианте, когда программы на школьном алгоритмическом языке записывались на доске и в тетрадях. Этот язык с национальной лексикой для записи алгоритмов был предложен академиком А.П.Ершовым [1]. Компьютерной альтернативой школьному алгоритмическому языку предполагался разработанный в Новосибирском Университете и Сибирском отделении АН СССР язык Рапира. В это же время на механико-математическом факультете МГУ им. М. В. Ломоносова был разработан редактор-компилятор «Е-практикум», позволяющий решать задачи из учебников на компьютерах, используя школьный алгоритмический язык. Впоследствии практикум был существенно доработан, например, в него были включены виртуальные исполнители «Робот» и «Чертежник» [2], появившиеся в новых вариантах учебников по информатике. Эта цифровая образовательная среда носит имя «КуМир», имеет широкое распространение, работает на всех настольных компьютерах, и эта разработка сейчас поддерживается и свободно распространяется ФНЦ НИИСИ РАН [3,4].

Постановка проблемы

Последние годы всё больше внимания направлено на дистанционные формы обучения. Массовые открытые онлайн-курсы, пик популярности которых пришёлся на второе десятилетие 21 века, также требуют включение цифровых образовательных сред в контент. Необходимы такие средства, как автоматическая проверка заданий, проверка на антиплагиат и многое другое. Для подобных систем целесообразно использовать серверные решения, предоставляя обучающимся доступ к заданиям и тестирующим системам через интернет. Кроме того, веб-приложения не нужно устанавливать на свой компьютер, тем самым ученики не привязаны к конкретному рабочему месту и могут продолжать выполнять задания в транспорте, по дороге домой, используя смартфоны и планшеты с доступом в интернет.

Цифровая образовательная среда КуМир является мультиплатформенным приложением и требует установки на компьютер учени-

ка[5]. При создании системы были разработаны средства индивидуальной проверки выполненных заданий, более того, задания было возможно объединять в единый тематический урок-занятие — практикум. Практикумы содержали набор задач с описанием и встроенными средствами проверки правильности выполненных задач (тесты также подготавливались на школьном алгоритмическом языке, и были скрыты от ученика). Тем не менее для учёта успешно выполненного практикума требовался непосредственный контакт с учителем, фиксирующим факт успешного решения или отправка «на проверку» учителю так называемой тетради с выполненными заданиями практикума. Для ученика последнее, иногда, становилось серьёзной задачей. Кроме того, ученик в качестве подтверждения выполнения работы мог отправить учителю чужую тетрадь, что сложно было проверить, так как КуМир не предоставлял средств однозначной идентификации для тетрадей практикума. Все эти проблемы возможно решить проведя интеграцию ЦОС КуМир в образовательную платформу Мирера, аналогично интеграции ЦОС ПиктоМир, ЦОС ЭВМ-практикум [6].

Основная часть

Первый шаг интеграции в ЦОП Мирера [7] состоял в доработке компилятора КуМир для встраивания в Миреру. Если школьник или студент выполнял задания по другим курсам в Мирере, то он имел собственный аккаунт, позволяющий ему подгружать от своего имени тетради с решениями в соответствующий контест (практикум) в Миреру. Одновременно, средствами ЦОП, проводилась проверка на заимствование решения (иди части решения) от другого ученика в текущем курсе или ранее загруженных в других курсах.

Второй шаг состоял в возможности создавать и проверять решения на школьном алгоритмическом языке непосредственно в ЦОП Мирера.

Для подготовки курсов педагогами было создано множество задач, в том числе наследованных из ранее подготовленных практикумов, снабжая каждую задачу эталонным решением и набором тестов. Необходимо указать, что встроенный в Миреру редактор не только сохранил раскраску программы в стиле КуМира, но и позволил подготавливать шаблоны задач, делая часть программного кода не редактируемым, аналогично средствам ЦОС КуМир, что методически сильно упрощает процесс обучения, особенно на начальном шаге. За-

дачи с графическими исполнителями Робот и Чертёжник также возможно проверять отдельно от практикума.

Выводы

Для ведения курсов по алгоритмике и программированию на факультете начального образования в МПГУ было подготовлено более 500 задач с решениями непосредственно в ЦОП Мирера. Однако, этого недостаточно.

Для удобства учеников необходимо визуализировать результаты работы программы на школьном алгоритмическом языке с исполнителями Робот и Чертёжник, аналогично решениям с производственными компиляторами по курсам Qt+OpenGL, Python и др. В этом случае ученик видит конечный результат работы программы и сможет проводить отладку программы непосредственно в ЦОП Мирера, аналогично неграфическим заданиям на школьном алгоритмическом языке.

Кроме того, сейчас проводится доработка ЦОС КуМир с добавлением аутентификации пользователя, позволяющая однозначно идентифицировать работу ученика в ЦОС Кумир и автоматически учитывать результаты выполнения им практикумов с отметкой в ЦОП Мирера. Аналогичная задача была уже ранее решена при интеграции ЦОС ЭВМ-практикум и платформы Мирера [6]. Работая с программой ЭВМ-практикум, ученик автоматически отправлял свои результаты прохождения курса в Мирерy, либо мог решать задачи непосредственно в ЦОП Мирера через интернет.

Заключение

Безусловно, редактор-компилятор КуМир позволяет ученику эффективнее проводить отладку программы при пошаговом выполнении с визуализацией содержания и результатов, или сразу видеть сообщения о синтаксических ошибках на полях программы [8]. Встраивание элементов визуализации содержания переменных в редактор Мирерy может стать новой целью развития интеграции ЦОП Мирерy и ЦОС КуМир.

Работа выполнена в рамках госзадания FNEF-2022-0010 «Разработка, реализация и внедрение семейства интегрированных многоязыковых сред программирования с автоматизированной проверкой заданий для учащихся образовательных организаций, ДОО, младшей, основной и старшей школы и студентов педагогических университетов».

Литература

[1] Ершов А. П. Алгоритмический язык. — М.: Квант, 1980. — Т. 1. — С. 42–45.

[2] Леонов А.Г., Первин Ю.А., Зайдельман Я.Н. Программные исполнители в цифровых образовательных средах «ПиктоМир», «Роботландия» и «КуМир». Информатика в школе. 2019;(9):54-61. https://doi.org/10.32517/2221-1993-2019-18-9-54-61

[3] Поляков К. Ю., Ерёмин Е. А. Информатика 7 — 9 классы // Методическое пособие. — М.: БИНОМ. Лаборатория знании, 2016. — 80 с.

[4] Рабочая программа. Методические комментарии: учебно-методическое пособие / А. Г. Кушниренко, А. Г. Леонов, Я. Н. Зайдельман, В. В. Тарасова. — Дрофа Москва, 2017. — 88 с.

[5] Стартовая страница проекта «КуМир» на сайте ФГУ ФНЦ НИИСИ РАН. URL: https://www.niisi.ru/kumir/ [Дата обращения 01.04.2022]

[6] Леонов А. Г., Орловский А. Е. Методы интеграции цифровых образовательных сред в цифровую образовательную платформу Мирера // Труды НИИСИ РАН. Математическое и компьютерное моделирование сложных систем: теоретические и прикладные аспекты. — 2021. — Т. 11, № 3. — С. 59–65.

[7] Леонов А. Г., Бесшапошников Н. О., Прилипко А. А. Цифровизация образования — Новые возможности управления образовательными треками // Вестник кибернетики. — 2018. — Т. 30, № 3. — С. 154–161.

[8] Леонов А. Г., Первин Ю. А. Элементы программирования в непрерывном курсе школьной информатики // Ярославский педагогический вестник журн. — 2013. — Т. 3, № 1. — С. 45–50.

И. Е. Панченко
Москва, Postgres Professional
Проект: Postgress Pro

Новое в учебных проектах Postgres Pro

Аннотация

Вкратце изложу состояние наших проектов по преподаванию СУБД в высшем и дополнительном образовании; расскажу о нашем видении развития, включая непрерывную программу подготовки системных разработчиков, а может быть, и создание под неё целого ИТ-факультета. Эти темы спорные, неоднозначные, и хотелось бы обсудить их с участниками конференции.

Е. Р. Алексеев, канд. тех. наук, доцент, С. П. Грушевский, доктор пед. наук., профессор
Краснодар, Кубанский Государственный Университет
math.kubsu.ru, 1aer.ru

Опыт использования свободного программного обеспечения при подготовке учителей математики и информатики в Кубанском Государственном Университете

Аннотация

Представлен опыт использования свободного программного обеспечения на кафедре информационных образовательных технологий Кубанского Государственного Университета при подготовке учителей математики и информатики.

Информационный блок дисциплин занимает заметное место при обучении будущих учителей математики и информатики. С 2018 года свободное программное обеспечение значительно шире стало применяться на кафедре информационных образовательных технологий (ИТО) Кубанского государственного университета при подготовке будущих учителей математики и информатики (направление подготовки — 44.03.05 Педагогическое образование (с двумя профилями подготовки), направленность — Математика, информатика).

Подготовка бакалавров на кафедре ИТО осуществляется 5 лет (10 семестров). Информационные дисциплины будущие учителя математики и информатики изучают с первого семестра до последнего.

На *первом курсе* студенты изучают «Программирование», «Программное обеспечение ЭВМ».

Курс программирования рассчитан на два года. Студенты знакомятся с алгоритмизацией и языками программирования Pascal, C/C++, Python. В лабораторных работах по программированию на Pascal используются FreePascal, Lazarus, PascalABC. Блок «Программирование на C/C++» ориентирован на использование компиляторов gcc, g++ и сред программирования Geany, Code::Blocks. Кроме того в теме «Вывод графической информации» используется Gnuplot. В блоках по программированию на Паскале и C/C++ используются курсы, разработанные одним из авторов по заказу компании Базальт СПО и представленного сайте `https://kurs.basealt.ru/`. В блоке «Программирование на Python» используются среды разработки IDLE, Spyder, библиотеки numpy, scipy, matplotlib, sympy. Во время изучения этого курса некоторые студенты получают индивидуальные задания, связанные с технологиями программирования на базе свободного ПО.

Стандартный курс «Программное обеспечение ЭВМ» посвящён знакомству студентов с операционными системами, офисными приложениями, языкам разметки Markdown, HTML. Кроме того, будущие учителя математики и информатики знакомятся с LaTeX. Отдельный блок посвящён первому знакомству с Linux. В качестве программного обеспечения используются: один из современных дистрибутивов Linux, LibreOffice, LaTeX, TexStudio.

Кроме того во втором семестре в курсе «Введение в направление подготовки» читаются лекции, посвящённых отечественным процессорам Байкал, Эльбрус и операционным системам Астра Линукс, Альт и Роса.

На *втором курсе* студенты изучают курс, посвящённым математическим пакетам, продолжают изучать программирование. Кроме того знакомятся с визуальным проектированием приложений в курсе «Визуальное объектно-ориентированное программирование». Курс «Математические пакеты и их применение в естественно-научном образовании» предполагает знакомство со Scilab, Octave, Maxima и Cantor. Курс «Визуальное объектно-ориентированное программирование» ещё предстоит переводить на свободное ПО.

На *третьем курсе* в дисциплинах «Технологии web-программирования», «Web-проектирование и web-дизайн» студенты углублённо изучают HTML, CSS, знакомятся с JavaScript. В качестве программного обеспечения выступают web-браузеры Mozilla FireFox, Seamonkey, современные текстовые редакторы и web-редактор Bluefish.

На *четвертом курсе* будущие учителя изучают следующие предметы информационного профиля «Теоретические основы информатики», «Компьютерная графика», «Операционные системы и компьютерные сети». Стандартный для учителей информатики курс «Теоретические основы информатики» предполагает выполнение лабораторных работ с применением программирования на любом изученном языке программирования. Могут использоваться программные средства FreePascal, Lazarus, gcc, g++, Geany, Code::Blocks, IDLE, Spyder, numpy, scipy, matplotlib, sympy. В курсе компьютерной графики студентами на практических и лабораторных занятиях используются свободные программы GIMP, LibreCAD, Blender. Курс «Операционные системы и компьютерные сети» преподаётся в 8 и 9-м семестрах. В восьмом семестре, наряду с общей теорией операционных систем, на практических занятиях студенты продолжают более глубокое знакомство с ОС Linux. Во второй части курса будущие учителя, наряду с общей теорией современных компьютерных сетей, самостоятельно настраивают локальную сеть в консольном режиме. Среди лабораторных работ можно выделить следующие: настройка и использование FTP-сервера, клиента и сервера ssh, настройка и использование samba. В качестве свободного программного решения может выступать любой современный дистрибутив Linux.

На *пятом курсе* студенты продолжают изучать «Операционные системы и компьютерные сети», а также в программе предусмотрен интенсивный курс компьютерного моделирования (9 недель, 3 часа лекций, 3 часа лабораторных работ в неделю). На лабораторных работах курса «Компьютерное моделирование» студенты используют свободные математические пакеты Scilab, Octave, а также свободные компиляторы и интерпретаторы.

Было написано несколько *курсовых работ*, посвящённых разным аспектам использования свободного программного обеспечения учителями.

В *магистратуре* (направление 01.04.01 «Математика», направленность «Преподавание математики и информатики») свободное про-

граммное обеспечение широко используется в следующих дисциплинах информационного блока: «Компьютерные технологии в науке и образовании», «Математические основы курса информатики», «Построение и использование свободных операционных систем в науке и образовании». В курсе «Компьютерные технологии в науке и образовании» магистранты широко изучают вопросы, связанные с использованием свободных математических пакетов (Scilab, Octave, Maxima) в учебном процессе и при решении реальных инженерных и научных задач. Курс «Математические основы курса информатики» является продолжением бакалаврского курса «Теоретические основы информатики». В магистратуре будущие учителя разрабатывают уроки по информатике для школы и лабораторные работы по информационным дисциплинам для высших учебных заведений. В последние два года большое внимание уделяется разработке уроков и лабораторных работ с использованием свободного программного обеспечения. Начиная с 2021–22 учебного в магистерскую программу будущих преподавателей включён курс «Построение и использование свободных операционных систем в науке и образовании». Это курс включает в себя следующие блоки:

- использование операционных систем семейства Linux на уроках информатики в школе;
- использование операционных систем семейства Linux на лабораторных работах в дисциплинах информационного профиля в средних специальных и высших учебных заведениях;
- практикум по сборке специализированных дистрибутивов операционных систем семейства Linux.

В следующем году планируется добавить блок, посвящённый отечественным операционным системам и их использованию в образовании.

За последние два года написано несколько *выпускных работ*, посвящённых использованию свободного программного обеспечению в образовании. Среди можно выделить следующие:

1. Свободные системы компьютерной математики при изучении производной в курсе математического анализа (Мандрыкина Дарья)
2. Разработка программного обеспечения для генерации образовательного дистрибутива (Бондаренко Роман).

В этом году планируется защита выпускных работ.

1. Параллельные и конвейерные вычисления в курсе «Технологии программирования» (магистерская диссертация, Мандрыкина Дарья).
2. Разработка лабораторного практикума «Основы работы в Linux» (выпускная бакалаврская работа, Номоконова Юлия)

В учебном процессе активно используются учебники и учебные пособия по свободному программному обеспечению, изданные в компании «ALT Linux». На базе них планируется издание методических пособий по отдельных дисциплинам.

Работа выполнена при финансовой поддержке Кубанского научного фонда в рамках научного проекта № ППН-21.1/10 «Цифровая дидактика для предметного обучения, воспитательной работы учащихся и профессиональной подготовки учителей».

Диденко Денис Владимирович
Щёлково, Государственное бюджетное профессиональное образовательное учреждение Московской области «Щёлковский колледж»

Использование свободного программного обеспечения в ГБПОУ МО Щёлковский Колледж СП-5

Аннотация

В докладе рассмотрен успешный опыт применения свободного программного обеспечения в ГБПОУ МО «Щёлковский колледж» СП-5, не только в образовательных целях, но и в работе специалистов.

Общие сведения

В ГБПОУ МО «Щёлковский колледж», структурном подразделении №5 успешно используются следующие программные продукты из линейки свободного программного обеспечения:

1. Операционная система Альт Образование 9 — Компьютерный класс из 14 ПК с принтером и сканером.
2. Офисный пакет LibreOffice

a) LibreOffice Writer — данный редактор используется при подготовке ВКР. Имея мощную систему стилей и плагинов он составляет достойную конкуренцию текстовому процессору от Microsoft.

b) LibreOffice Calc — данное приложение используется не только на занятиях по информационным технологиям, но и в работе ряда специалистов. Например, наш психолог использует данное приложение для автоматизации необходимых психологических тестов и вывода результатов на печать. Несколько лет назад, совместно с Академией социального управления МО и Центром Практической Психологии Образования в рамках Московской области был запущен проект по диагностике явлений буллинга в подростковой среде. Изначально специалистами центра была разработана достаточно обширная бумажная методика, но мы предложили её автоматизировать с условием — только в LibreOffice (официально по юридическим соображениям). Месяц работы психологами Подмосковья был в итоге достаточно успешен, хотя первичное использование программы происходило местами с затруднением.

c) LibreOffice Impress — данный редактор в представлении не нуждается — стандартный редактор презентаций. У нас в колледже некоторые преподаватели далёкие от IT пользуются им и LibreOffice в целом.

d) СУБД LibreOffice Base — кроме изучения баз данных на уроках информатики, что на сегодняшний день достаточно тривиально, наша психологическая служба использует данную СУБД для ведения своего электронного журнала.

3. Bluefish — это приложение используется для работы с вебстраницами.

4. CMS — системы управления содержимым сайта

   a) CMS Wordpress

   b) CMS Joomla!

5. Filezilla — двухпанельный файловый менеджер. В основном используется для передачи файлов сайта на хостинг.

6. LibreCAD — система автоматизированного проектирования (САПР), построенная на лицензии GNU/GPL — используется в обучении строительных специальностей, таких как:

   a) 08.02.01 Строительство и эксплуатации зданий и сооружений;

   b) 08.02.11 Управление, эксплуатация и обслуживание многоквартирного дома;

   c) 08.02.08 Монтаж и эксплуатация оборудования и систем газоснабжения.

   На сегодняшний день, после известных всем событий в сфере программного обеспечения данное приложение используется не только в нашем структурном подразделении, но и в других, так как всем известная компания Autodesk закрыла доступ Российским студентам к своим программным продуктам.

7. Gimp — аналог всем известного Adobe Photoshop. В представлении данный редактор не нуждается. В профессиональном направлении он используется при подготовке графики для сайтов и при разработке дизайн-макетов сайтов.

8. Veyon — достойное приложение для администрирования компьютерного класса.

9. Inkscape — данный редактор мы используем для изучения векторной графики и для подготовки графики для сайтов.

10. Kdenlive — с помощью данного приложения мы изучаем основы нелинейного видеомонтажа.

Организация работы с ОС на базе ядра Линукс в СП 5 ГБПОУ МО «Щёлковский Колледж»

Все свободные приложения успешно используются в операционной системе «Альт образование 9» и миграция с данных программных продуктов на другие (например, в производственной деятельности) не вызывает затруднений, так как основная часть инструментов использующихся в данных приложениях идентичная.

Выводы

По результатам апробации операционной системы Альт Образование был сделан однозначный вывод, что процесс обучения в колледже довольно успешно реализуется на основе операционной системы на базе ядра Линукс, с использованием свободного программного обеспечения, включённого в состав дистрибутива (или доустановленного из репозитория).

Большинство используемых проприетарных программ имеют свободные или бесплатные аналоги, которые имеют Линукс-версии и позволяют их использовать в работе для решения каждодневных задач пользователей, что для небольших предприятий с ограниченным бюджетом очень большое подспорье.

Кроме того, в условиях государственных образовательных организаций, которые, с одной стороны, обладают достаточно скудным бюджетом, а с другой обязаны соблюдать лицензионное законодательство, всё это позволяет эффективно решать вопросы не только финансового, но и учебного характеров.

Клепалов Александр Владимирович
Нижний Тагил, Муниципальное автономное общеобразовательное учреждение средняя общеобразовательная школа №100

«Организация доменной инфраструктуры в общеобразовательной школе на базе решений Базальт СПО»

Аннотация

В докладе описывается опыт организации домена в ЛВС общеобразовательной школы. Приводится информация о структуре домена, операционных системах, используемых на ПК входящих в домен, программном обеспечении, используемом в процессе управления, дополнительном ПО, расширяющем функциональность контроллера домена. Также приведена информация о мерах по обеспечению информационной безопасности, резервирования данных и перспективах развития доменной инфраструктуры

Муниципальное автономное общеобразовательное учреждение средняя общеобразовательная школа №100, была построена в 2019

году. Оснащение школы обучающими материалами и компьютерной техникой соответствовало действующим на тот момент стандартам. В связи с этим компьютерный парк школы превышает 500 единиц. Для централизованного управления таким количеством техники возникла необходимость создания домена.

Ввиду неизбежного перехода государственных и муниципальных организаций на отечественное ПО, в качестве основы доменной инфраструктуры была выбрана ОС «Альт Сервер». Средствами данной ОС, был создан домен `Samba`, являющийся аналогом `Active Directory`.

Первоначально в составе домена работало 48 ПК с ОС «Альт Образование». После заключения договора о сотрудничестве с Базальт СПО и получения по этому договору дополнительных лицензий на Альт Образование, количество ПК в домене было увеличено до 168.

Для ПК, не используемых непосредственно в учебном процессе, были куплены лицензии Альт Рабочая станция и Альт 8 СП (сертификат ФСТЭК). ПК с Альт Рабочая станция были сразу без проблем интегрированы в домен. С ОС Альт 8 СП возникли проблемы с работой групповых политик, исправленные после очередного обновления. Таким образом, на текущий момент, в состав домена входит 186 ПК с разными дистрибутивами ОС Альт.

Поддержка групповых политик, реализованная в Альт Сервер, облегчает процесс настройки прав и рабочего окружения пользователей домена. В частности, активно используются инструменты установки домашней страницы браузеров, создания ярлыков программ, правил подключения usb-дисков, параметров удалённого доступа, и установки дополнительного ПО.

В связи с тем, что большая часть пользователей ПК, на которых установлены ОС Альт — это школьники, которые часто очень склонны ко всяким экспериментам с настройками техники, которой они пользуются, самыми востребованными в школе групповыми политиками являются те, что задают ограничения на изменение графического оформления системы — обоев, заставки, темы и так далее. Но на данный момент, эти политики влияют только на ПК, с графической оболочкой Mate. В ОС «Альт Образование», установленной на ученические ПК, используются графические оболочки XFCE и KDE Plasma. Поэтому ведутся поиски альтернативных методов блокировки настроек рабочей среды на ученических ПК.

Для управления доменом используется ОС «Альт Сервер 10» с установленной утилитой ADMC, и windows 7 с RSAT. Утилиты имеют схожий интерфейс, и внутреннюю логику, что очень упрощает их совместное использование. Возможности, заложенные в ADMC на данный момент перекрывают большую часть задач, возникающих в повседневной работе с доменом: создание/редактирование пользователей и компьютеров, настройка общих папок и так далее. RSAT используется для настройки групповых политик, в той части, которая пока не реализована средствами ADMC.

Поскольку почти две трети компьютерного парка школы — это ноутбуки, весьма важным вопросом является ограничение доступа в сеть посторонних устройств. Никакие пароли не обеспечивают надёжную защиту доступа в сеть. Школьники начиная с 7-8 классов прекрасно осведомлены о способах подбора пароля — например, с помощью известной утилиты Reaver. Первоначально ограничение доступа обеспечивалось фильтрацией по белому списку mac-адресов. Но исчерпание ёмкости таблицы фильтра, вынудило перейти на другие способы аутентификации. В этом вопросе очень помогло наличие в Альт Сервер встроенного сервера Radius.

Помимо авторизации пользователей в ОС, доменная авторизация используется в других программных комплексах, используемых в школе: системе облачного хранилища Nextcloud, системе управления компьютерным классом Veyon. В случае с Nextcloud, сквозная авторизация облегчает работу пользователей — нет необходимости запоминать различные учётные данные для ПК и облака. Использование доменной авторизации в Veyon, упрощает управление каталогом клиентских компьютеров, и обеспечивает более очевидную идентификацию управляемых ПК на компьютере учителя.

Сейчас для школы закуплено ещё 360 лицензий Альт Образование, для замены ОС Windows на практически 100% ПК. Такое увеличение количества компьютеров в домене, влечёт за собой обновление технической части контроллера домена, в частности, переноса контроллера с обычного ПК, на стоечный сервер с существенно большими аппаратными ресурсами. Также, планируется введение вторичного контроллера.

По мере увеличения функциональности встроенных в ОС Альт средств управления доменом, в частности ADMC и GPUI, есть желание отказаться от использования связки windows 7 и RSAT.

Литература

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

Владимир Башун, Антон Сергеев
Москва, Московский институт электроники и математики им. А.Н. Тихонова НИУ ВШЭ
https://github.com/wekan/wekan,
https://github.com/taigaio/taiga-back,
https://github.com/taigaio/taiga-front

Система управления проектной работой студентов как драйвер развития цифровой экосистемы университета

Аннотация

В связи с массовым внедрением проектной модели обучения в Московском институте электроники и математики им. А. Н. Тихонова НИУ ВШЭ появилась необходимость разработки системы проектного управления, позволяющей автоматизировать основные операции контроля и менеджмента работы студенческих команд: вести онлайн-мониторинг активности, интенсивности разработки, движения кадрового состава при записи и отчислении с проектов, оценивать эффективность руководителей и т.п.

Цель системы — сделать процесс проектной работы максимально прозрачным для всех сторон и получить набор показателей для объективной оценки успешности проектов. В работе представлена архитектура, ключевые технические особенности, механизмы встраивания в цифровую инфраструктуру Университета. Отдельное внимание уделено вопросам интеграции Системы с другими цифровыми сервисами проектной экосистемы МИЭМ и НИУ ВШЭ: трекерами, Gitlab, облачными хранилищами, системами ВКС и т.п. Рассмотрен вопрос работы с корпоративной SSO, позволяющей осуществлять бесшовные переходы между внешними и локальными (wekan, taiga, ВКС и др.) сервисами.

Работа в формате реализации проекта под задачи государственных и бизнес-структур является, пожалуй, самым результативным и эффективным методом организации подготовки кадров в высшей

школе. Актуальные задачи от предприятий реального сектора экономики, междисциплинарность, навыки командной работы и, главное, конкретные и востребованные результаты такой работы — всё это делает проектную модель обучения востребованной для студентов, бизнеса и образовательных организаций.

Студенты, сталкиваясь с реальными задачами и жёсткими требованиями проектной работы, могут на практике применить багаж знаний и навыков, приобретённый во время обучения, опробовать свои силы на «боевых» задачах. Компании могут проверить силами студенческих команд самые смелые гипотезы (proof-of-concept), провести анализ перспективных направлений, получить действующий прототип новых систем и, главное, сплочённую и проверенную в деле команду молодых специалистов. Вуз получает прямые связи с индустрией, поток актуальных задач, рост уровня подготовки студентов и квалификации преподавателей.

Выпускники, имеющие реальный опыт решения практических задач в конкретной предметной области, обладают для работодателя неоспоримыми преимуществами: наличие необходимых знаний и навыков, быстрое погружение в трудовую деятельность, сокращённый период интеграции в существующие процессы и др.

Московский институт электроники и математики (МИЭМ) им. А. Н. Тихонова НИУ ВШЭ с 2018 года перешёл на проектную модель обучения: 100% студентов, начиная со 2 курса, проходят через дисциплину «Проект», связанную с реализацией задач индустриальных партнёров Университета. Обучение построено так, чтобы проект стал центром траектории обучения студента, вокруг которого строится основная образовательная программа. Это помогает сформировать профессиональные навыки, как «жесткие» (hard skills), так и «мягкие» (soft skills), и лучше подготовить студентов к работе в бизнес-среде.

Постановка задачи: система управления проектной работой студентов

Безусловно, масштабное внедрение проектной модели обучения в вузе связано со значительными трудностями и, в первую очередь, управленческими. «Как учитывать эффективность и результаты проектов?», «Как контролировать ход работ сотен проектов, в которые вовлечены тысячи студентов?», «Учебный план и проект — как све-

сти их вместе?», «Как передавать результаты проектной работы от выпускников к новой студенческой команде?».

В связи с этим возникла необходимость создать единую экосистему проектной работы, которая позволила бы гибко вести все рабочие процессы, связанные проектно учебной деятельностью: от записи в проекты (через систему вакансий), отслеживания прогресса работы, прохождения контрольных точек и учёта реальных трудозатрат.

Проект принципиально отличается от курсовой работы или ВКР тем, что это прежде всего командная работа, нацеленная на достижение чёткого результата, обозначенного в техническом задании. Как несомненный бонус, участие в проектах должно приучать студентов к лучшим индустриальным практикам работы, включая такие практики как совместная работа над исходным кодом, учёт и трекинг задач в корпоративной среде, умение ставить задачи и контролировать их выполнение, соблюдение сроков разработки, гибкие (agile) технологии проектного управления, использование Scrum/Agile/Kanban и т.п. Система так же должна:

- предоставить аналитические и статистические данные;
- поддерживать интеграцию с внешними сервисами;
- быть модульной и расширяемой;
- сделать все процессы работы над проектом прозрачными и открытыми.

С точки зрения вуза, важным является необходимость контролировать не только студентов, но и руководителей проектов (категории сотрудников ППС или АУП). И, конечно, важным является стоимостной фактов: бюджет решения, традиционно, сильно ограничен.

Экосистема проектной работы

Необходимо выстроить единую экосистему, которая позволит решить указанные задачи. При этом для технического вуза, большая часть проектов составляют программные и программно-аппаратные, что и обуславливает выбор элементов системы.

Ключевые элементы этой системы:

1. Система управления репозиториями программного кода.
2. Инструмент поддержки жизненного цикла DevOps.
3. Система учёта затраченного времени (трекинг задач).

4. Средства для организации видеоконференций — для обеспечения коммуникации в режиме удалённой работы.
5. Средства корпоративной коммуникации (email, корпоративный мессенджер).
6. Единая система авторизации и управления доступом.
7. Система накопления документации на базе wiki.
8. Система «цифрового следа», накапливающая информацию по работе в проектах из всех возможных источников.
9. Единая точка входа во все сервисы (кабинет проектной работы).

Для любого вуза вопрос бюджета на разработку стоит остро, поэтому при построении единой цифровой экосистемы, преимущество отдаётся выбору свободно распространяемого программного обеспечения.

Архитектура системы представлена на рис. 1.

В качестве элементов системы, где это было возможно, использовались готовые решения с открытым исходным кодам. При этом отдавалось предпочтение on-premise решениям, которые могут быть развёрнуты на серверах организации. Использование облачных или внешних сервисов практиковалось, но имело ряд недостатков. К примеру, организация столкнулась со следующими проблемами:

1. В качестве системы учёта затраченного времени и трекинга задач изначально использовался бесплатный сервис Trello. Однако, постоянные изменения API и политики использования, которые в итоге сделали невозможным его использование на бесплатной основе, привели к необходимости поиска аналогов.
2. Для сервиса видеоконференций в НИУ ВШЭ были закуплены лицензии Zoom, однако в связи с ужесточением санкционного режима компания отказалась продлевать лицензию. К счастью, в МИЭМ к этому моменту уже было разработано альтернативное решение.

Используемые решения и технологии

В качестве продуктов сопровождения проектной деятельности, были выбраны следующие:

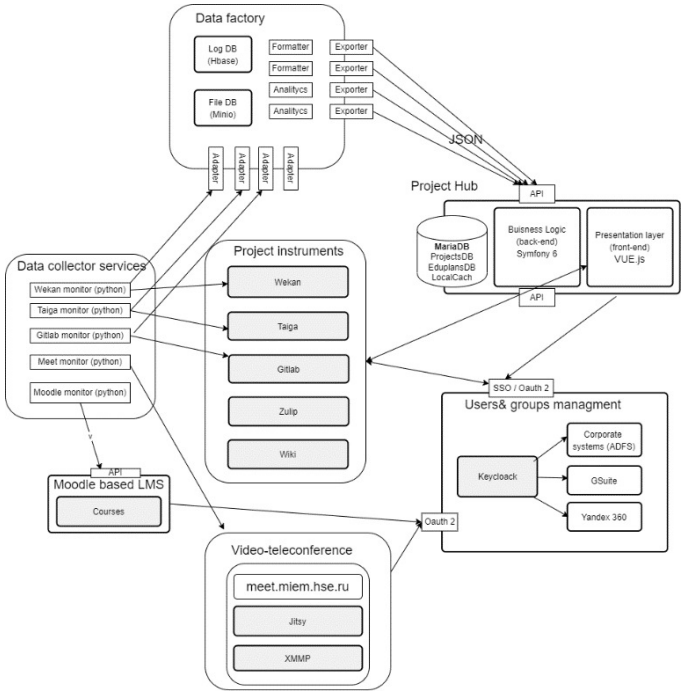


Рис. 1: Архитектура системы.

1. Для системы управления репозиториями программного кода и инструмент поддержки жизненного цикла DevOps — self-managed версия системы Gitlab [1].
2. Для системы управления проектом и учёта затраченного времени (трекинг задач) — предоставлены альтернативы:
   a) Простой Kanban — сервис wekan [2]
   b) Agile/Scrum/Kanban с расширенным функционалом — сервис taiga [3,4]
3. Средства для организации видеоконференций — был в качестве основы взят self-hosted сервис jitsi. Сервис был доработан для удобства использования, встраивания и сбора статистики (`https://meet.miem.hse.ru`).

4. Средства корпоративной коммуникации — используется корпоративный мессенджер zulip [6].
5. Единая система авторизации и управления доступом — развёрнут SSO сервер keycloak [7].
6. Система накопления документации на базе Wiki JS [8]
7. Система «цифрового следа» — реализовано собственное решение для системы хранения BigData.
8. Единая точка входа во все сервисы (кабинет проектной работы) — реализовано собственное решение, которое объединяет все указанные выше сервисы и реализует логику проектного обучения (регистрация проектов, запись на проект и т.п.) [9].

Разработанная платформа внедрена в МИЭМ НИУ ВШЭ и успешно эксплуатируется более 2х лет.

В целом, успешная разработка и внедрение Системы управления проектами (`cabinet.miem.hse.ru/`позволяет 2-м менеджерам проектного офиса МИЭМ НИУ ВШЭ управлять до 500 проектов в год, в которые вовлечены до 2000 студентов, 200 руководителей проектов и 40 организаций-заказчиков, среди которых Huawei, ГК АстраЛинукс, ГК Инфовотч, ISS, Webinar, ES Leasing, Тинькофф, институты РАН и др.

Литература

[1] Инструмент поддержки жизненного цикла DevOps `https://about.gitlab.com/install/`

[2] Система учёта затраченного времени `https://github.com/wekan/wekan`

[3] Система управления проектом и учёта затраченного времени `https://www.taiga.io/`

[4] Средства для организации видеоконференций `https://github.com/jitsi`

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

[6] Корпоративный мессенджер `https://github.com/zulip/zulip`

[7] Система авторизации и управления доступом `https://github.com/keycloak/keycloak`

[8] Система wiki https://js.wiki/

[9] Система поддержки учебно-проектной деятельности https://cabinet.miem.hse.ru/

Старых Владимир Александрович
Москва, НИУ ВШЭ
https://miem.hse.ru/edu/ce/

Внедрение свободного ПО на примере пакета Альт в инфраструктуру инженерных образовательных программ университета МИЭМ НИУ ВШЭ

Аннотация

Российские операционные системы являются неотъемлемой составляющей процесса импортозамещения. Они активно используются в государственном секторе, где предъявляются особые требования к защите и обработке информации, стабильности и совместимости систем и используемых приложений. Российские ОС позволяют реализовать весь комплекс задач без дополнительных вложений в разработку и модернизацию.

ДКИ МИЭМ НИУ ВШЭ следит за развитием отечественных операционных систем и поддерживает партнёрские отношения с разработчиками, которые уже завоевали доверие и доказали на практике эффективность и успешность представленных на рынке продуктов. Наш многолетний опыт работы гарантирует, что ваша задача будет решена с помощью проверенных и актуальных решений. Альт линукс — многофункциональная операционная система, построенная на ядре Linux и семейство дистрибутивов с поддерживаемой целостностью. Дистрибутивы легко интегрируются между собой позволяя выстраивать корпоративные платформы и системы любого масштаба. Функциональность продуктов позволяет наладить привычные рабочие процессы без использования на предприятии зарубежного или не включённого в реестр программного обеспечения. Все продукты и программные решения, представленные командой ALT Linux Team, направлены на работу с конфиденциальной информацией, лицензированы ФСТЭК и включены в Единый реестр российских программ для ЭВМ.

В образовательных программах департамента компьютерной инженерии МИЭМ НИУ ВШЭ используются дистрибутивы Альт Линукс построенные на отечественной инфраструктуре Сизиф (`Sisyphus`). Это российский репозиторий, находящийся под юрисдикцией РФ, который успешно поддерживается командой Alt Linux Team. Философия `Sisyphus` заключается в технологии сборки программ и учёта зависимостей между ними. В учебном процессе и проектном обучении также активно используются такие программные продукты, как Альт Рабочая станция, ОС Альт 8 СП, Альт Сервер и Альт Север Виртуализации.

Мария Петрова
Санкт-Петербург, ООО «Базальт СПО»

Диверсификация сертификации: конец уравниловки

Аннотация

В докладе будут представлены направления развития системы сертификации специалистов по продуктам и решениям «Базальт СПО». Каким образом студентам университетов или колледжей подтвердить свой профессиональный уровень в области отечественных операционных систем? Открытый проект «Альт Академия» предлагает своим участникам такую возможность. Для тех, кто изучает ОС «Альт» в рамках основного учебного плана или факультатива, будут открыты сертификационные испытания. Их успешное прохождение позволит подтвердить знания и навыки сетевого и системного администрирования Linux-систем, дополнив профессиональное портфолио сертификатом ведущего разработчика отечественного ПО. На конференции мы планируем обсудить с педагогическим сообществом перспективы, потенциал и способы организации студенческой сертификации.

За последние два года система подготовки специалистов, поддерживающих решения на базе операционных систем семейства «Альт», существенно изменилась качественно и количественно. Сформирована линейка авторизованных учебных курсов, растёт число сертифицированных преподавателей, количество администраторов, успешно освоивших работу с дистрибутивами «Базальт СПО» в рамках повышения квалификации достигло тысячи, программы образовательных учреждений разного уровня всё чаще основываются на ОС «Альт».

За этими достижениями логично следует развитие системы сертификации специалистов по продуктам и решениям «Базальт СПО». Мы предлагаем несколько уровней сертификационных испытаний в зависимости от степени подготовленности и профессиональных целей кандидатов.

Первая ступень сертификации будет предоставлять студентам вузов и колледжей, изучающим ОС «Альт» в рамках основной или дополнительной образовательной программы, возможность подтвердить весомость полученного багажа знаний, поместив в профессиональное портфолио сертификат ведущего разработчика отечественного ПО и облегчив свой карьерный старт.

Эта же ступень сертификации будет рекомендована специалистам технической поддержки первой линии. Им будут предложены базовые сертификационные испытания, успешное прохождение которых позволит партнёрам «Базальт СПО» организовать надёжную систему технической помощи с опорой на собственный штат компетентных сотрудников. Результатом подобных испытаний станет присвоение статуса сертифицированного Администратора ОС «Альт».

Следующую ступень сертификации мы разработали для тех, кто реализует проекты по переводу ИТ-инфраструктуры на операционные системы семейства «Альт», а затем поддерживает работоспособность и устойчивость таких решений. В этом случае испытания будут проходить в два этапа: предварительное тестирование и выполнение стендового задания. Эта ступень соответствует статусу сертифицированного Специалиста по продуктам и решениям «Базальт СПО».

Те специалисты, кто занимается разработкой IT-проектов на уровне предприятия или региона, смогут подтвердить свои знания, получив сертификат самой высокой ступени. Для этого потребуется продемонстрировать умение планировать технологические решения по успешному внедрению отечественного ПО, обеспечивающие гибкую интеграции в действующую информационную структуру, информационную безопасность, комплексную поддержку всех отраслевых производственных процессов.

Совершенствованию структуры сертификации специалистов по продуктам и решениям «Базальт СПО» будет способствовать сотрудничество компании с Международной «Ассоциацией специалистов по сертификации», которое развивается с 2021 года.

Ступени сертификации разрабатываются с учётом возможности подготовки к ним на базе авторизованных учебных курсов «Базальт

СПО» и программ переподготовки кадров. Приобретение необходимых для прохождения сертификационных испытаний компетенций студентами университетов и колледжей может быть включено в основную образовательную программу по ИТ-дисциплинам.

Виктор Минченков, Антон Сергеев, Владимир Башун
Москва, Московский институт электроники и математики им А.Н. Тихонова НИУ ВШЭ
https://git.miem.hse.ru/vminchenkov/meet-front

Масштабируемая система видеоконференций для корпоративного использования

Аннотация

Переход многих компаний и учебных учреждений на удалённый формат работы остро поднял вопросы по организации коммуникации между работниками. При этом существующие видеосервисы для многих не всегда закрывают потребности образовательных организаций, дороги при использовании полного функционала (в расчёте на пользователя), не соответствуют требованиям корпоративных политик безопасности. Есть сложности с их интеграцией в существующую цифровую экосистему вузов. Столкнувшийся с этими вызовами МИЭМ НИУ ВШЭ принял решение разработать и внедрить собственную корпоративную ВСК на базе движка с открытым кодом Jitsi. Время показало, что в условиях пандемии, онлайн-работы и внешних санкционных ограничений, это было единственно верным решением. В статье представлена система онлайн-коммуникаций MEET.MIEM, её архитектура, используемые технологии, методы интеграции в корпоративную цифровую инфраструктуру.

В начале 2020 года институт МИЭМ ВШЭ столкнулся со сложностями в организации общения между студентами и преподавателями, так как начался активный переход студентов в дистанционный формат проведения занятий и развитие проектного формата обучения. Дополнительную нагрузку создавала внедрённая в вузе 100%-проектная модель обучения: проектные студенческие команды (более 400 проектов по 2-6 человек) должны были проводить онлайн-совещания не менее 1 раза в неделю. Организация закупки лицензий на такое количество пользователей (2000+) было не целесообразно по причине стоимости и невозможности полной интеграции в цифровую

проектную экосистему. Было принято решение развернуть систему видео конференции с возможностью бесплатного использования на собственных мощностях.

Выбор базовой платформы

Из свободных для использования решений на рынке в тот мент можно было выделить два решения BigBlueButton (BBB) и Jitsi[1]. Оба решения базировались на протоколе WebRTC и могли работать без установки дополнительного ПО на стороне пользователей, используя только современный браузер.

Для окончательного внедрения были выбраны следующие критерии по функциональности:

1. Поддержка от 20 и более параллельных конференций;
2. Поддержка авторизации пользователей;
3. Возможность масштабирования решения;
4. Поддержка конференции до 60 человек;
5. Простота использования для пользователей;
6. Возможность модификации исходного кода для дальнейшего наращивания функционала;
7. Наличие мобильного клиента для iOS и Android.

После быстрого сравнения систем было установлено, что BBB почти не обладал возможностями по масштабированию и требовал гораздо больших аппаратных ресурсов при большом количестве параллельных конференций.

Наращивание функционала

В итоге тестирования базовая платформа Jitsi была выбрана для внедрения и использования для учебного процесса.

После использования в течение трёх месяцев в системе были выявлены следующие серьёзные недостатки, усложняющие жизнь нашим пользователям:

1. Система поддерживала только парольную систему авторизации и могла быть интегрирована только с каталогами LDAP. В самом НИУ ВШЭ внедрена корпоративная система ведения учётных записей студентов и преподавателей на базе платформы

Microsoft AD, и система SSO ADFS и её подключение к Jitsi было невозможно;

2. Парольная система авторизации, используемая в Jitsi, блокировалась современными антивирусами, из-за чего пользователи иногда не могли присоединяться к конференциям.

3. Полностью отсутствовала возможность сбора статистических данных по пользователям, так как дистанционный формат обучения требовал дополнительного контроля;

4. Отсутствие централизованного хранения и публикации видео записей встреч.

Детальное изучение платформы показало, что в её основе лежит XMPP[2] сервер PROSODY [1] для которого можно писать дополнительные модули на языке LUA, а сам режим работы системы авторизации пользователей можно перевести JSON Web Tokens (JWT). Сама медиа-платформа JITSI meet имеет два хорошо документированных API для встраивания в сторонние Web приложения.

Эти два факта позволили создать собственную точку входа (Web приложение) для пользователей конференций MEET.MIEM, и конечная реализация использует следующую архитектуру (на рисунке 1):

1. Frontend: web приложение на базе Vue 2.0 и JS фреймворка Bootstrap-VUE;

2. Backend: Python WEB фреймворк FLASK 2.0. Данный фреймворк позволил сразу использовать JWT для авторизации пользователей через стороннюю SSO KEYCLOAK по протоколу OAuth2;

3. Database: В качестве базы данных может быть использована PostgreSQL или MariaDB;

4. В качестве дополнительного сервиса была осуществлена поддержка системы Sentry для своевременного сбора данных об ошибках на Frontend и Backend и формы обратной связи от пользователей;

5. Сама медиа платформа Jitsi-meet была интегрирована во frontend через свой Iframe API и также через подключение авторизации через JWT продолжилась совместимость нашей платформы с мобильными приложениями Jitsi для всех платформ;

6. Реализован дополнительный модуль для PROSODY, позволяющий экспортировать внутренние события в конференциях;
7. Для сервисов Jibri (дополнительный сервис, реализующий потоковую запись конференций Jitsi) был реализован модуль позволяющий выгружать видеофайлы в систему хранения, а на backend отправлять событие, что запись создана. При этом ссылки на видео сохраняются в БД, для последующего доступа пользователями.

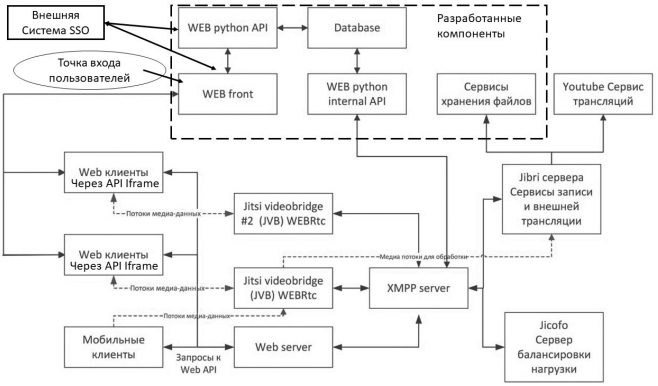


Рис. 1: Итоговая внедрённая архитектура системы видеоконференций

Ключевым ограничением MEET.MIEM (как и аналогичных систем, построенных на Jitsi, например, Яндекс.Телемост) является лимит на число пользователей в одной комнате (не более 60). Причина — ограничения браузеров по организации потоковых соединений.

Заключение

Разработка и внедрение собственной системы видеоконференций закрыло большинство требований по организации удалённой работы студентов и преподавателей, позволило централизованно накапливать и экспортировать данные о проведении занятий студентов. И, в итоге, стало фундаментом для налаживания нормальной работы Института в онлайн-режиме в условиях карантинных ограничений.

На базе платформы MEET.MIEM проводятся массовые онлайн мероприятия, например, Техношоу МИЭМ `https://technoshow.miem.hse.ru/` или Финал национального чемпионата WorldSkills Russia по одной из основных компетенций. Поддерживаемая пиковая нагрузка системы, развёрнутой на одной серверной платформе CPU 2xXEON 128Гбайт RAM при выделенном канале 0,5 Гбит/с составляет до 200 онлайн-пользователей одновременно.

После расширения функционала силами МИЭМ у участников появилась возможность удобного ведения и просмотра истории своих конференций в том числе просмотра видео записей встреч и чата внутренних сообщений. Дополнительное внедрение системы пропусков позволило преподавателям удобно подключать к встречам других участников, не имеющих постоянных аккаунтов в НИУ ВШЭ.

Литература

[1] Проект Jitsi. `https://github.com/jitsi`

[2] XMPP сервер Prosody. `https://prosody.im`

Е.,Р. Алексеев, канд. тех. наук, доцент, К. В. Дога, студентка, Ю. Н. Номоконова, студентка
Краснодар, Кубанский Государственный Университет
1aer.ru

Учебное пособие «Первое знакомство с ОС Linux»

Аннотация

Представлено учебное пособие «Первое знакомство с ОС Linux», разработанное в Кубанском Государственном Университете. На платформе stepik пособие представлено в виде онлайн-курса. Пособие активно используется на факультете математики и компьютерных наук Кубанского Государственного Университета.

На сегодняшний день на факультете математики и компьютерных наук (ФМиКН) Кубанского Государственного Университета (КубГУ) студенты изучают Linux на разных курсах и направлениях. Кроме того, студенты и преподаватели используют различные дистрибутивы Linux. Возникла необходимость создания универсального учебного

пособия как для самостоятельного знакомства, так и для внедрения в программу различных информационных дисциплин.

Пособие предназначено:

- для знакомства с такими понятиями как: операционная система, ядро, дистрибутив;
- для изучения общих принципов функционирования и использования операционных систем семейства Linux;
- получения практического опыта работы с командной строкой в Linux.

Пособие имеет практически нулевой порог вхождения. Такие сложные термины, как «операционная система», «ядро операционной системы», «дистрибутив» объясняются начинающему пользователю на доступном языке. При этом новые определения являются полноценными.

Цели пособия:

- познакомить учащегося с основными теоретическими сведениями об операционных системах семейства Linux;
- при выполнении лабораторных работ приобрести необходимые практические навыки работы в Linux.

Пособие состоит из 8 частей и приложения

1. Знакомство с ОС Linux
2. Файловая система ОС Linux
3. Терминал Linux и основные команды
4. Создание нового пользователя. Изменение прав доступа
5. Пакеты в ОС Linux
6. Настройка начального загрузчика
7. Жёсткие и символьные ссылки
8. Установка ОС Linux

В приложении приведены 8 лабораторных работ (15 вариантов) от простых заданий к более сложным.

Следует отметить, что пособие не ориентировано на работу с конкретным дистрибутивом. По возможности авторы пытались сделать его универсальным. Начинающий пользователь учится работать в консоли, причём как с deb, так и rpm пакетами. Базовые команды

описаны так, чтобы учебное пособие могло использоваться как справочник. Более сложные пользователь сможет найти самостоятельно благодаря командам получения справочной информации, о которых также подробно рассказано. Большое внимание авторы уделили практическим задачам, которые помогут закрепить навыки, необходимые при ежедневной работе с Linux. Пособие распространяется свободно по лицензии GNU FDL и доступно на сайте `http://1aer.ru/` и на Яндекс диске по адресу `https://disk.yandex.ru/i/BtJF1pwoFUEg8g`.

Также на платформе stepik (по ссылке `https://stepik.org/course/111193/syllabus`) эта работа выставлена в виде курса, причём помимо существующих лабораторных работ, были разработаны дополнительные тестовые задания различных видов для лучшего усвоения и закрепления теоретических знаний.

Сейчас пособие активно используется на факультете математики и компьютерных наук Кубанского Государственного Университета:

1. В курсе «Программное обеспечение ЭВМ» на первом курсе направления подготовки «Педагогическое образование (с двумя профилями подготовки). Преподавание математики и информатики» при изучении раздела «Основы работы в ОС Linux»
2. В курсе «Технологии программирования и работы на ЭВМ» направления «Математика и компьютерные науки» для приобретения практических навыков работы в Linux.
3. В магистерских курсах информационного профиля для быстрого самостоятельного освоения операционных систем семейства Linux.
4. В качестве базового пособия по основам Linux в магистерском курсе «Построение и использование свободных операционных систем в науке и образовании»
5. В качестве практикума «Основы работы в Linux» в дисциплине «Введение в направление подготовки» на первом курсе направления «Педагогическое образование (с двумя профилями подготовки). Преподавание математики и информатики»
6. В курсе «Операционные системы и компьютерные сети» направления подготовки «Педагогическое образование (с двумя профилями подготовки). Преподавание математики и информатики» для быстрого самостоятельного освоения операционных систем семейства Linux.

Работа выполнена при финансовой поддержке Кубанского научного фонда в рамках научного проекта № ППН-21.1/10 «Цифровая дидактика для предметного обучения, воспитательной работы учащихся и профессиональной подготовки учителей».

Е. Р. Алексеев, канд. тех. наук, доцент, Д. С. Мандрыкина, магистрант
Краснодар, Кубанский государственный университет
1aer.ru

Использование свободных компиляторов при изучении технологий параллельного программирования

Аннотация

Разработан модуль «Технологии параллельного программирования» базового курса по программированию направления «Математика и компьютерные науки». Модуль включает в себя курс лекций и лабораторный практикум. Лабораторный практикум разработан на базе свободных компиляторов gcc, g++, gfortran

В связи с ростом производительности программного обеспечения, IT-специалисты должны знать технологии ускорения вычислений. Обучение параллельному программированию — это одна из главных задач в нынешней подготовке программистов. В КубГУ студенты направления «Математика и компьютерные науки» знакомятся с параллельным программированием в процессе изучения курса «Технологии программирования и работы на ЭВМ». Основная цель дисциплины «Технологии программирования и работы на ЭВМ» — подготовка студентов в области применения современных технологий программирования и вычислительной техники в решении прикладных задач, связанных с обработкой данных, математическим моделированием, созданием программного обеспечения системного и прикладного уровня[1]. Курс охватывает основные алгоритмы, которые нужно знать и понимать студентам инженерных специальностей. В модуле «Параллельное программирование» рассматриваются современные средства разработки кроссплатформенных параллельных приложений, а также особенности организации параллельных вычислений в многоядер-

ных и многопроцессорных вычислительных системах. Модуль включает в себя курс лекций и лабораторный практикум по следующим темам:

- разработка параллельных программ с использованием библиотеки MPI;
- разработка параллельных программ с использованием библиотеки OpenMP;
- особенности разработки параллельных программ в современном Фортране.

Лабораторный практикум основан на работе в свободных компиляторах семейства gcc (gcc, g++, gfortran).

На лабораторных работах студенты разрабатывают параллельные программы и сравнивают время их выполнения с временем выполнения последовательных программ.

Были разработаны инструкции по установке свободных компиляторов на личные компьютеры студентов и методические рекомендации по разработке параллельных приложений на компьютерах в лаборатории и на личных ноутбуках студентов.

Курс предусматривает разработку нескольких параллельных математических программ[4]:

- операции с матрицами;
- численное интегрирование;
- решение систем линейных алгебраических уравнений.

Отдельное место в блоке параллельного программирования уделено разработке программ на Фортране. Современный Фортран имеет большое количество подпрограмм для работы с матрицами, что ускоряет как написание кода, так и время его выполнения за счёт встроенной оптимизации[5].

К примеру, рассмотрим время выполнения алгоритма умножения матриц (1000x1000), написанного на языках C++ и Fortran. MATMUL(matrix_a, matrix_b) — встроенная функция Фортрана для матричного умножения[7]. Представленные здесь и ниже результаты получены на ПК с характеристиками Intel Core i5-1135g7 с тактовой частотой 2.4 Ггц, 4 физических ядра (8 логических), 16 ГБ ОЗУ, SSD 512ГБ:

	Время выполнения
C++	4,02
Fortran	3,5
Fortran (matmul)	0,3

В разработанном курсе программы численного интегрирования рассматриваются при освоении параллельной технологии MPI. Студентам предлагается написать параллельную программу численного интегрирования одним из предложенных методов и протестировать её на разном количестве потоков. Работа может быть выполнена как на Фортране, так и на C++. Результаты выполнения программы интегрирования ($f(x) = \frac{x}{8+x^2}$, $n = 10^9$) представлены в таблице.

	np = 1	np = 2	np = 4	np = 8
g++	12.24	6.15	3.77	1.89
gfortran	3.47	1.75	1.07	1.07

Параллельное расширение OpenMP для языков C++ и Fortran помогает достичь ускорения за счёт распараллеливания практически без изменения обычного последовательного алгоритма. При изучении этой технологии студенты используют директивы #pragma omp parallel в последовательных алгоритмах своих программ и сравнивают полученные результаты времени выполнения.

Рассмотрим время выполнения последовательного и параллельного алгоритмов умножения матриц, полученных с помощью семейства компиляторов gcc.

	Последовательный алгоритм	Параллельное выполнение
g++	4.02	1.45
gfortran	3.5	2.18

Изучение модуля «Параллельное программирование» в системе подготовки бакалавров математического факультета Кубанского государственного университета является необходимым условием формирования предметной компетенции. Разработанный модуль позво-

лит реализовать подготовку студентов к реальным задачам, требующим параллельной обработки данных. Лабораторный практикум разработан на базе свободных компиляторов gcc, g++, gfortran. При обучении параллельному программированию свободные компиляторы помогают оценить существенное ускорение выполнения программ и понять принципы работы технологий параллельного программирования.

Были получены следующие результаты:

- разработан модуль «Параллельное программирование» для подготовки бакалавров Математического факультета КубГУ в рамках изучения дисциплины «Технологии программирования»;
- написаны лекции по параллельному программированию;
- разработаны лабораторные работы по параллельному программированию;
- предложены инструкции по установке и использованию свободных компиляторов на личных компьютерах;
- написаны методические рекомендации по разработке параллельных приложений на компьютерах в лабораториях ФМиКН КубГУ и на личных ноутбуках студентов;
- продемонстрирована эффективность использования технологий параллельного программирования и обосновано использование свободных компиляторов в учебном процессе;
- все материалы находятся в свободном доступе на сайте 1aer.ru и включены в обучающий курс со свободным доступом на образовательной платформе Stepik.org — https://stepik.org/112412

Работа выполнена при финансовой поддержке Кубанского научного фонда в рамках научного проекта № ППН-21.1/10 «Цифровая дидактика для предметного обучения, воспитательной работы учащихся и профессиональной подготовки учителей».

Литература

[1] Алексеев Е. Р. Опыт использования свободного программного обеспечения в курсах «Операционные системы» и «Параллельные вычисления» / Е. Р. Алексеев, Н. Ю. Болтачева, Д. А. Лутошкин // Общество, наука, инновации (НПК-2016) : Сборник статей 2-е издание, исправленное и дополненное, Киров, 18–29 апреля 2016 года / Вятский государственный

университет. — Киров: Вятский государственный университет, 2016. — С. 2664–2669.

[2] Шеметова А. Д. Методические приёмы обучения параллельному программированию/ Прикладная информатика. — 2016. — Т. 11. — № 6(66). С. 43 — 48. ISBN 978-5-04050-682-8.

[3] Гергель В. П. Современные языки и технологии параллельного программирования. — М.: МГУ, 2012. — 408с.

[4] Борзунов С. В., Кургалин С. Д., Флегель А. В. Практикум по параллельному программированию. — СПб: БХВ, 2017. — 236.

[5] Свободные и проприетарные компиляторы C(C++) и Фортрана при разработке эффективных вычислительных приложений / Е. Р. Алексеев, П. А. Дёмин, Д. А. Лутошкин, В. В. Стародумов // Современные информационные технологии и ИТ-образование. — 2017. — Т. 13. — № 4. — С. 232–240. — DOI 10.25559/SITITO.2017.4.446.

[6] Алексеев Е. Р. Новые технологии разработки высокоэффективных и параллельных приложений на современном Фортране / Е. Р. Алексеев, П. А. Дёмин, Н. Ю. Болтачева // Прикладная информатика. — 2018. — Т. 13. — № 1(73). — С. 103–120.

[7] Алексеев Е. Р. Современный язык программирования Фортран в образовании и научных исследованиях / Е. Р. Алексеев, О. В. Соболева // Современные информационные технологии и ИТ-образование. — 2016. — Т. 12. — № 4. — С. 110–116.

Алексеев Е. Р., канд. тех. наук, доцент, Гончаров С. В., магистрант
Краснодар, Кубанский Государственный Университет
1aer.ru

Свободные библиотеки интервальных вычислений при подготовке бакалавров и магистров направления «Математика и компьютерные науки» в Кубанском государственном университете

Аннотация

Изучение интервальных вычислений входит в программу подготовки бакалавров и магистров направления «Математика и компьютерные науки» в Кубанском Государственном Университете. Бакалавры знакомятся с интервальным анализом в курсе «Современные методы обработки числовых данных». Магистранты продолжают изучение интервальных вычислений в рамках курса «Компьютерные технологии в науке и образовании». Используя свободные библиотеки интервальных вычислений, авторы разработали библиотеку решения инженерных задач, которая используется в учебном процессе.

Увеличение вычислительных мощностей современных компьютеров предоставляет возможность решать сложные научные и инженерные задачи, нередко требующие большого количества операций с плавающей точкой. Решение задач, описывающих реальные высокоточные процессы — есть хороший фундамент подготовки будущих инженеров и программистов.

Одним из средств повышения достоверности компьютерных вычислений является интервальный анализ. В Кубанском Государственном Университете изучение интервальных вычислений входит в программу подготовки бакалавров и магистров направления «Математика и компьютерные науки». Бакалавры направления подготовки 02.03.01 «Математика и компьютерные науки» (направленность подготовки «Вычислительные, программные, информационные системы и компьютерные технологии») знакомятся с интервальным анализом в курсе «Современные методы обработки числовых данных». Магистранты (направление подготовки 02.04.01 Математика и компьютерные науки, направленность — Вычислительная математика) продолжают изучение интервальных вычислений в рамках курса «Компьютерные технологии в науке и образовании».

На лабораторных работах используются следующие свободные программные средства интервальных вычислений[1]:

1. Interval (`https://sourceforge.net/p/octave/interval/ci/default/tree/`) — пакет интервальных вычислений для Octave.
2. Boost interval (`https://github.com/boostorg/interval`) — библиотека интервалов на C++.
3. Libieeep1788 (`https://github.com/boostorg/interval`) — библиотека интервалов, реализующая интервальную арифметику стандарта IEEE 1788.

В бакалавриате студенты в курсе «Современные методы обработки числовых данных» решают стандартные задачи, связанные с высокоточными вычислениями, а также несложные задачи интервального анализа (найти корни нелинейного уравнения, задачи линейной алгебры). Для решения подобных задач большинство студентов пользуется пакетом интервальных вычислений в Octave.

В рамках лабораторных работ при изучении интервальных вычислений магистрантам могут быть предложены задачи, имеющие реальное прикладное инженерное значение. Например,

1. Нестационарная задача теплопроводности [2]

$$\begin{gathered}\frac{\partial t(x,\tau)}{\partial \tau} = a\frac{\partial^2 t(x,\tau)}{\partial x^2};\\ (\tau > 0); 0 \le x \le \delta;\\ t(x,0) = t_0;\\ \frac{\partial t(0,\tau)}{\partial x} = 0;\\ t(\delta,\tau) = t_{\tilde{n}\grave{o}}.\end{gathered}$$

2. Задача Ван-Дер-Поля [3]

$$\begin{gathered}\frac{dy_1}{dt} = y_2\\ \frac{dy_2}{dt} = \mu(1-y_1^2)y_2 - y_1;\\ y_1(0) = 2;\\ y_2(0) = 0;\\ 0 \le t \le T\end{gathered}$$

   где μ — коэффициент жёсткости. Причём, чем больше значение μ, тем более жёсткой становится задача.

3. Прямая и обратная геодезические задачи [4]

   Прямая задача: по заданной широте $\varphi 1$ и долготе $\lambda 1$ первой точки требуется найти географические координаты $\varphi 2$ и $\lambda 2$ второй точки, если известны начальный азимут α и расстояние D между этими точками (рисунок 1).

   Обратная задача: найти расстояние D между двумя заданными точками и азимут α из первой точки на вторую.

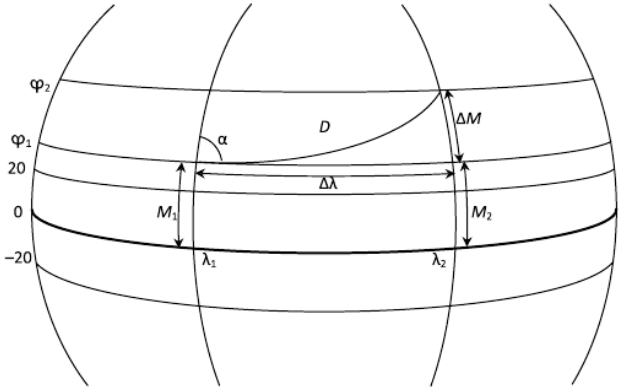


Рис. 1: Графическая модель прямой геодезической задачи

4. Жёсткое дифференциальное уравнение первого порядка [5]

$$\varepsilon y' + (1+t)y - t - 1 = 0;$$
$$y(0) = 0; t \in (0, 2)$$

5. Дифференциальное уравнение [5]

$$y' + y\tan(t) - 100t^2\cos(t)^2 = 0;$$
$$y(0) = 0; t \in (0, \frac{\pi}{2})$$

6. Краевая задача [5]

$$\varepsilon y'' + xy' + y = 0;$$
$$y(0) = 1;$$
$$y(1) = \exp(-0.5\varepsilon);$$
$$\varepsilon = 1/300$$

Количество часов, отводимых на изучение интервальных задач в магистратуре в рамках курса «Компьютерные технологии в науке и образовании» невелико. Чтобы все магистранты смогли справится с реальными задачами, была разработана библиотека решения задач

методом интервальных вычислений. Библиотека использует класс интервалов библиотеки boost interval и свободные компиляторы семейства gcc. В состав библиотеки входят функции, реализующие:

1. Простейшие матричные операции (сложение, вычитание, умножение матрицы на число, произведение матриц, транспонирование матриц, вычисление обратной матрицы), вычисление определителя матрицы.
2. Алгоритмы решения систем линейных алгебраических уравнений (метод Гауса, Жордана-Гауса, LU-разложение, QR-разложение).
3. Алгоритмы решения нелинейных уравнений и систем.
4. Алгоритмы решения дифференциальных уравнений первого и второго порядков.

Набор этих инструментов позволит использовать библиотеку в решении моделей инженерных задач в образовательном процессе.

В результате изучения интервальных вычислений в рамках дисциплин «Современные методы обработки числовых данных» и «Компьютерные технологии в науке и образовании» студенты получат не только теоретические знания, но и смогут применить эти знания на практике при решении прикладных инженерных задач.

Работа выполнена при финансовой поддержке Кубанского научного фонда в рамках научного проекта № ППН-21.1/10 «Цифровая дидактика для предметного обучения, воспитательной работы учащихся и профессиональной подготовки учителей».

Литература

[1] Интервальный анализ и его приложения. Программное обеспечение и языки программирования. url: `http://www.nsc.ru/interval/?page=Programing`

[2] Ерёмин А. В., Кудинов И. В. Об одном методе решения нестационарных задач теплопроводности Вестник Самарского государственного технического университета. Серия: Технические науки. 2012. № 2 (34). С. 158–164.

[3] Хайрер Э., Ваннер Г. Решение обыкновенных дифференциальных уравнений. Жёсткие и дифференциально-алгебраические задачи: пер. с англ. М. : Мир, 1999. 685 с.

[4] Ботнев В. А., Устинов С. М. Методы решения прямой и обратной геодезических задач с высокой точностью Научно-технические ведомости Санкт-Петербургского государственного политехнического университета. Информатика. Телекоммуникации. Управление. 2014. № 3 (198). С. 49–58.

[5] Бандурин Н. Г. Численное решение жёстких нелинейных задач. Вестник Волгоградского государственного архитектурно-строительного университета. Серия: Строительство и архитектура. 2010. № 17 (36). С. 17–23.

Волканов Дмитрий Юрьевич, Курячий Георгий Владимирович
Москва, факультет ВМК МГУ имени М. В. Ломоносова

Цепочка кафедральных курсов, использующих свободное программное обеспечение

Аннотация

В докладе рассказывается про опыт преподавания курсов «Язык программирования Python», «Совместная разработка на Python», «Разработка программного обеспечения для GNU/Linux», «Архитектура современных ЭВМ» и «Практические аспекты сетевых протоколов в Linux» на кафедре АСВК факультета ВМК МГУ имени М.В. Ломоносова и рассказывается о том, какое свободное программное обеспечение используется в рамках данных курсов.

Основным направлением деятельности кафедры АСВК [1] факультета ВМК МГУ имени М. В. Ломоносова является подготовка специалистов по разработке распределённой инфраструктуры для передачи больших потоков данных. В настоящее время разработан ряд курсов, выстраивающихся в единую цепочку [2]. В докладе рассказывается о тех курсах, что читались в 2021/2022 учебном году.

На младших курсах студенты факультета ВМК изучают Паскаль, Ассемблер, C и C++. Студенту при выполнении курсовой работы необходимо владеть языком программирования Python. В рамках курса «Язык программирования Python» изучают современное состояние языка программирования Python, синтаксис и основные приёмы программирования на нём, а также выполняют домашние задания. В рамках курса используется Python и его инфраструктура.

Разработка программ это не только написание кода, но и ряд дополнительных активностей, а также умение работать в команде. На курсе «Совместная разработка приложений на Python» изучаются три аспекта совместной разработки приложений: инструменты и дисциплина совместной работы (DVCS), направления разработки (документирование, тестирование, интернационализация, деплоймент), а также инструментарий, определяемый языком программирования Python. В качестве отчёта студенты объединяются в микрогруппы и пишут совместный проект. В курсе рассматриваются система контроля версий git, средство документирования кода Sphinx, несколько систем локализации и сборки.

Разработка невозможна без знания современных архитектур. Архитектура процессоров RISC-V на данный момент, наиболее стройная и понятная из всех актуальных архитектур вычислительных систем. В курсе «Архитектура и язык ассемблера RISC-V» прослеживается связь между конкретной организацией процессора и общими принципами построения вычислительных систем. В учебном процессе используются эмулятор RARS, а также среда разработки и визуализатор выполнения кода RISC-V.

Специалист по разработке должен понимать, как работает сеть Интернет. Задача курса «Практические аспекты сетевых протоколов в Linux» иллюстрация организации и работы семейства сетевых протоколов TCP/IP на примере операционной системы семейства GNU/Linux. В курсе рассматриваются все уровни TCP/IP в той мере, в какой они используются системой. Для практических занятий в курсе предусматривается использование виртуальных машин VitrualBox.

Разработка невозможна без знания жизненного цикла создания программного обеспечения (ПО). В курсе Разработка программного обеспечения для GNU/Linux рассматривается жизненный цикл создания программного продукта на примере практик, используемых при разработке приложений на C для операционной системы семейства GNU/Linux.

Всё ПО используемое в рамках курсов является свободным и обладает такими свойствами, как публичная доступность, достаточная для обучения функциональность, возможность лицензирования и публикации результатов. Несвободное ПО имеет ряд усложнений, затрудняющее его изучение.

Также стоит отметить, что все лекции читаются в прямом эфире на сайте youtube.com, все желающие могут задать вопросы в чате и затем лекции выкладываются в открытом доступе.

Литература

[1] Сайт кафедры АСВК — https://asvk.cs.msu.ru/

[2] Сайт сообщества UNИХ — http://uneex.ru/

Наталья Ковалёва
Москва, Московский педагогический государственный университет (МПГУ)

Использование свободного программного обеспечения в программе технической направленности организации дополнительного образования

Аннотация

В статье автором рассмотрены принципы креативного обучения, которые необходимо использовать в программе дополнительного образования технической направленности. Для реализации принципов предложены программные решения «Базальт СПО». С их помощью предлагается выстроить систему подготовки детей и формирования творческих способностей для последующего обучения профессиональным навыкам.

При построении системы образования на макроуровне и решении задач по формированию компетенций будущих специалистов важная роль отводится системе дополнительного образования детей. Дополнительное образование обеспечивает необходимые условия для личностного развития. Именно в дополнительном образовании раскрываются творческие способности детей, осуществляется поддержка и продвижение талантов.

Творческие возможности человека или креативность — это способности, которые проявляются в мышлении, чувствах, отдельных видах деятельности. Они характеризуют личность в целом и её отдельные стороны, продукты деятельности, процесс их создания, определяют её

готовность изменяться, отказываться от стереотипов, помогают находить оригинальные решения сложных проблем в ситуации неопределённости.

Креативность в профессиональной педагогике рассматривается как способность к творчеству, принятию и созданию нового, нестандартному мышлению, генерированию большого числа оригинальных и полезных идей [1].

При проектировании программ дополнительного образования, особенно технической направленности, обязательно нужно использовать креативное обучение, которое базируется на следующих принципах:

- принцип соответствия внешнего образовательного продукта учащегося его внутренним потребностям;
- принцип индивидуальной образовательной траектории учащегося в образовательном пространстве;
- принцип интерактивности занятий, осуществляемый при помощи средств коммуникаций и инструментов цифровых технологий;
- принцип открытой коммуникации по отношению к создаваемой учащимся образовательной продукции [2,3].

Обучение по дополнительным общеразвивающим программам технической направленности предполагает развитие творческого мышления, воображения, интуиции. Программы должны быть ориентированы на развитие интереса к техническому моделированию, привлечение обучающихся к современным технологиям конструирования и программирования. Техническая направленность обучения базируется на цифровых технологиях, что способствует удовлетворению потребностей общества в воспитании информационно грамотной личности.

Для эффективной реализации методов креативной работы и накопления креативного потенциала должны использоваться педагогические технологии:

- технология игровой деятельности,
- технология развивающего обучения,
- коммуникативная технология обучения,
- здоровьесберегающая технология.

Формами организации учебного занятия могут быть: дидактическая игра, конкурс, наблюдение, открытое занятие, практическое занятие, презентация, комбинированное занятие, творческие работы, самостоятельные работы репродуктивного характера, тестирование, защита творческих проектов, соревнование.

Профессиональное обучение детей школьного возраста с использованием предложенных технологий – это раннее развитие аналитического, образного и пространственного мышления у ребёнка, а также увеличение его конкурентоспособности в более зрелом возрасте на рынке труда.

На сегодняшний день существует много различных сред для обучения, созданных специально для детей. В современных реалиях необходимо ориентироваться на существующие на отечественном рынке программные решения, которые могут быть интегрированы как на единой платформе, так и востребованы как дискретные решения. Предложенные креативные методы и педагогические технологии можно реализовать с помощью комплекса разработок «Базальт СПО».

В качестве базовых решений для программы технической направленности организации дополнительного образования предлагается использовать:

1. На начальном уровне для вовлечения и знакомства с технологиями разработки и программирования `GCompris` — коллекцию обучающих игр и упражнений[1];

2. Для перехода к прикладному программированию и основам робототехники — среду Arduino IDE под управлением операционной системы Альт Образование 8[2];

3. При изучении алгоритмизации и технологии работы в средах разработки программного обеспечения — `Geany` и `Code::Blocks` под управлением операционной системы Альт Линукс (на примере Альт Образование 8)[3];

4. Для знакомства с основами инженерии и технического проектирования — программа твердотельного моделирования `FreeCAD`[4].

[1] `https://gcompris.net/index-ru.html`
[2] `https://kurs.basealt.ru/course/view.php?id=42`
[3] `https://kurs.basealt.ru/course/view.php?id=15`
[4] `https://kurs.basealt.ru/course/view.php?id=51`

Все эти программные продукты могут быть также использованы учащимися самостоятельно для продолжения обучения дома и при переходе на следующую ступень обучения.

Таким образом, работа с программными решениями «Базальт СПО» предполагает развитие проектного мышления для школьников, что является особенно востребованным и актуальным при формировании современных качеств личности. Всё это играет очень большую роль в дальнейшем развитии личности ребёнка и формировании его интереса к выбору будущей профессии.

Литература

[1] Башина, Т. Ф. Креативность как основа инновационной педагогической деятельности / Т. Ф. Башина. — Текст : непосредственный // Молодой учёный. — 2013. — № 4 (51). — с. 521–525. — URL: https://moluch.ru/archive/51/6639 / (дата обращения: 17.02.2022).

[2] Креативное мышление: как научиться мыслить нестандартно? URL: `https://sberuniversity.ru/upload/iblock/7b6/EduTech\_36\_web.pdf` (Дата обращения 17.02.2022)

[3] Николаева Е.И. «Психология детского творчества. 2-е изд.»: Питер; Санкт-Петербург; 2010 ISBN 978-5-49807-489-4

Евгений Ковалёв
Москва, Московский педагогический государственный университет (МПГУ)

Модель взаимодействия магистерской программы педагогического направления МПГУ и разработчика программного обеспечения «Базальт СПО»

Аннотация

В статье обосновывается необходимость взаимодействия компании-разработчика программного обеспечения и образовательной программы в вузе при организации учебного процесса в магистратуре. Автором рассматривается опыт института математики и информатики МПГУ и «Базальт СПО» при реализации подготовки магистрантов по программе «Проектирование цифровой среды образовательной организации».

Важный аспект при организации практико-ориентированного обучения будущих специалистов в высшей школе — налаживание взаимодействия образовательной организации и фирм-разработчиков программного обеспечения. Причём при выстраивании такого взаимодействия важно учитывать реализацию стратегии импортозамещения и использование отечественного программного обеспечения, входящего в реестр отечественного ПО. Таким образом, студенты получать необходимые знания и умения работы в прикладных профессиональных решениях, изучать аспекты правового применения программного обеспечения и будут готовы к проектам по комплексной цифровизации образовательной организации любого уровня сложности.

Перспективной формой реализации сотрудничества видится внедрение и поддержка спецкурса в рамках магистерской программы, а также его сопровождение фирмой-разработчиком при проведении различных форм учебных занятий, таких как, практики, научно-исследовательская работа и подготовка магистерской диссертации.

В институте математики и информатики МПГУ в рамках договора о сотрудничестве компания «Базальт СПО» и кафедра прикладной информатики и вычислительной математики, которая является выпускающей по магистерской программе 44.04.01 — Педагогическое образование, Проектирование цифровой среды образовательной организации, начали реализовывать модель взаимодействия, основанную на предложенных принципах.

Модель выстроена на базе курса «Технологии и инструменты цифровой образовательной среды», который изучается магистрантами во 2 учебном семестре. Базовые дисциплины, изученные магистрантами до этого, позволяют им не только разбираться с различных прикладным программным обеспечением, но и рассматривать задачу на построение единой цифровой среды, куда должны быть интегрированы инструменты по управлению организацией, анализу данных и внедрению заказных решений на базе программно-аппаратных платформ.

Модель направлена на формирование компетенций в области внедрения цифровых сервисов и цифровизации управления образовательной организацией на программных разработках «Базальт СПО». Основываясь на функционал существующей линейки разработок образовательных платформ «Базальт СПО», в качестве базового решения выбран пакет Альт Образование 10.

Модель предусматривает следующие шаги по её реализации:

1. регистрация и изучение дистанционного курса Альт Образование на портале `https://kurs.basealt.ru/`.
2. Обзор технологий и продуктов «Базальт СПО».
3. Изучение дистанционных курсов по выбранным направлениям цифровизации (ПО для управления ОО, ПО для цифровизации учебного процесса).
4. Проведение мастер-классов «Базальт СПО», изучение реализованных кейсов и опыта по цифровизации.
5. Прохождение практики и стажировки на базе «Базальт СПО», выбор дополнительных курсов или направлений изучения образовательных программных продуктов студентами.
6. Публикации результатов практики/стажировки, участие в студенческих научных и научно-методических конференциях.
7. Создание репозитория собственных разработок студентов.
8. Подготовка отчётов по научно-исследовательской работе студентов (НИРс), выбор и обоснование темы диссертации. Приглашение сотрудников и партнёров компании «Базальт СПО» для участия в интерактивных мероприятиях в качестве экспертов для внешней независимой оценки работ студентов.
9. Подготовка диссертационного исследования на базе собственной разработки и/или результатов практики.

Реализация предложенной модели позволит познакомить студентов с интегрированными решениями на базе СПО, а также позволит сформировать компетенции по цифровизации образовательной организации на основе опыта, накопленного компанией разработчиком.

Панюкова Александра Анатольевна
Москва, ГАПОУ КП№11, ООО «ШЭРИКС»
https://study.reversea.net/

Применение СПО в серии дисциплин основного, общего гуманитарного и социально-экономического цикла для IT-специалистов

Аннотация

Статья рассказывает о серии дисциплин, реализуемых на базе ГАПОУ КП №11, направленных на устойчивое формирование сообщества разработчиков, обладающих достаточными компетенциями для развития собственных проектов в области IT в целом и СПО в частности — «Информационная этика», блок «IT-решения» дисциплины «Инновационные технологии» и «Основы предпринимательства».

Рассмотрим серию курсов, построенных вокруг проектной деятельности, направленной на устойчивое формирование сообщества разработчиков, обладающих достаточными компетенциями для развития собственных проектов в области IT в целом и СПО в частности, реализуемых на базе ГАПОУ КП №11.

В учебной программе курсы называются «Информационная этика» и «Основы предпринимательства» (входят в профессиональный цикл как дисциплины общего гуманитарного и социально-экономического цикла) и раздел «IT-решения» курса «Инновационные технологии», который входит в профессиональный цикл основной профессиональной образовательной программы по ряду специальностей. Рабочие программы учебных дисциплин могут быть использованы в дополнительном профессиональном образовании при реализации программ повышения квалификации и профессиональной подготовки по профессии рабочих 16199 Оператор электронно-вычислительных и вычислительных машин.

Дисциплина «Информационная этика» введена с целью изучения моральных проблем, возникающих в связи с развитием и применением информационных технологий. Информационные технологии затрагивают фундаментальные права человека, касаясь защиты авторских прав, интеллектуальной свободы, ответственности и безопасности. Информационная этика рассматривает проблемы собственности,

доступа, неприкосновенности частной жизни, безопасности и общности информации. Рассматриваются проблемы коммуникации в ходе профессиональной деятельности и пути их решения. Учащиеся рассматривают вопросы формирования своего образа в сети, прорабатывая различные сценарии с позиции личности, наёмного сотрудника, фрилансера или основателя стартапа.

В результате освоения учебной дисциплины обучающийся должен уметь взаимодействовать со специалистами смежного профиля и корректно позиционировать себя и свою деятельность в информационном пространстве.

Именно на этом курсе многие учащиеся впервые осознают СПО не очередными программными продуктами, которые можно использовать, приняв какие-то условия не глядя, а концепцией, в пользу которой можно сделать осознанный выбор.

Программа курса построена без привязки к какому-то конкретному программному обеспечению и может быть реализована с применением любых устройств, на которых можно запустить браузер и любой простой текстовый редактор. Это даёт возможность проводить занятия в компьютерных классах с произвольным набором оборудования и ПО, позволяя развивать пользовательский кругозор обучающихся.

Последняя часть курса предполагает совместную работу над проектом, присутствующим в различных формах и проявлениях в двух других упомянутых ранее курсах — «Основы предпринимательства» и «IT-решения». Он, по задумке автора курса, должен являться продуктом, основанном на свободном решении. Однако, задумка автора встречает некоторое сопротивление в реальности, связанное с особенностями восприятия своей будущей профессии обучающимися и пониманием задачи в целом.

Дисциплина «Инновационные технологии» — это «дайджест» современных информационных технологий, позволяющих выпускнику ориентироваться в их мире. В разделе «IT-решения» студентам предлагается разделиться на команды разработчиков и принять участие в создании или доработке технического задания (и реализации прототипа или минимального жизнеспособного продукта — в зависимости от того, что позволяет время и навыки) проекта на основе имеющегося решения (разработка предыдущих поколений студентов).

В курсе «Основы предпринимательства» тот же проект рассматривается в другой плоскости — с точки зрения организации бизнеса в сфере IT. Как максимум, в результате курсов должно получиться

множество проектов на основе продукта с открытым исходным кодом. Прогресс из года в год есть, но значимый программный результат только начинает зарождаться. Потому что студенты не умеют писать код и читать чужой код и во время прохождения курсов только начинают понимать, зачем учиться это делать, и осознавать себя как будущих специалистов. Учатся организовывать свою работу, ответственности за свои действия и обещания.

Поскольку данные курсы глубоко не затрагивают компетенции разработки, отсутствие реализованного программного продукта на данном этапе нельзя назвать существенным недостатком концепции, поэтому она может быть применена и в других учебных заведениях, в том числе с формированием команд из разных образовательных организаций.

Все материалы без привязки к особенностям образовательного процесса и контингента обучающихся собираются автором на сайте проекта в отдельном курсе «Введение в sharing-сервисы», в котором на примере работы над проектом (тематика выбирается обучающимися в определённых рамках) рассматриваются все ключевые аспекты, заложенные в перечисленных ранее курсах. На его основе можно формировать материалы для иных образовательных организаций или групп людей, в том числе на основе других проектов, более подходящих потенциальной целевой аудитории.

Леонид Чашкин, Оксана Батонова, Кирилл Падалица, Елена Батракова, Стелла Скрыльникова, Полина Прилепко, Екатерина Прокофьева
Москва, НИУ ВШЭ
https://cabinet.miem.hse.ru/#/project/964

Оценка возможностей использования отечественной операционной системы Альт в развитии геоинформационных систем и технологий

Аннотация

Увеличение значимости и понимания глобальных экологических проблем ведёт к популяризации безопасности производства для окружающей среды и расширению применения чистых технологий на на-

циональном уровне, использование инновационных методов в управлении георисками и экологическом мониторинге. Особую важность здесь представляет ресурсный сектор, поскольку предприятия недропользования и энергетики составляют значимую долю производственной мощности и формирования бюджета страны. Большое значение здесь представляют программы «Интеллектуального горного производства», где управление осуществляется дистанционно или полностью автоматически. Другим важным вектором здесь является развитие технологий с учётом стратегии импортозамещения в сфере геоинформационных систем и технологий — в условиях её реализации становится важным исследовать возможности и преимущества отечественных операционных систем на примере дистрибутивов Alt на базе формирования технического приложения и обработки разноформатных данных геопространственного анализа по объектам открытых горных работ.

Введение

Перед использованием геопространственных данных их следует обработать. В том числе чтобы произвести анализ видео с дрона с последующей интеллектуальной обработкой, например, чтобы собрать базу данных для обучения нейронной сети, или чтобы дополнить карты местности 3d моделями, построить карту рельефа, требуется сначала произвести предобработку исходного материала.

Как уже упоминалось, существует несколько основных направлений, в которых происходит работа над исходным видеоматериалом для горнопромышленной сферы, а именно:

1. Создание базы данных либо для дальнейшего анализа, либо для использования в нейронных сетях глубокого обучения;
2. Интеграция с картами местности, где важнейшей задачей является сохранить привязку к пространственным координатам;
3. Удобная интерпретация информации для конечного пользователя (скорее, как промежуточный этап).

Описание решения

Для построения карты рельефа, полезных ископаемых и т.п., в первую очередь важно иметь привязку к пространственным координатам. Далее, при наличии избыточных или противоречащих друг другу данных, стоит избавиться от выбросов [1–6].

Опишем способ построения карты рельефа в свободнораспространяемой программе QGIS, входящий в состав дистрибутивов операционной системы «Альт».

Для привязки к пространственным координатам используется модуль QuickMapServices, позволяющий загрузить и работать с разными видами карт, в том числе OpenStreetMap. Модуль SRTM-Downloader используется для получения данных о высоте из проекта SRTM (Shuttle Radar Topography Mission). Программа QGIS позволяет на основе этих данных наложить на карту градиент, добавить изолинии, а затем построить карту рельефа (рис. 1). С помощью модуля qgis2threejs можно построить 3D-модель полученной карты рельефа для большей наглядности (рис. 2).

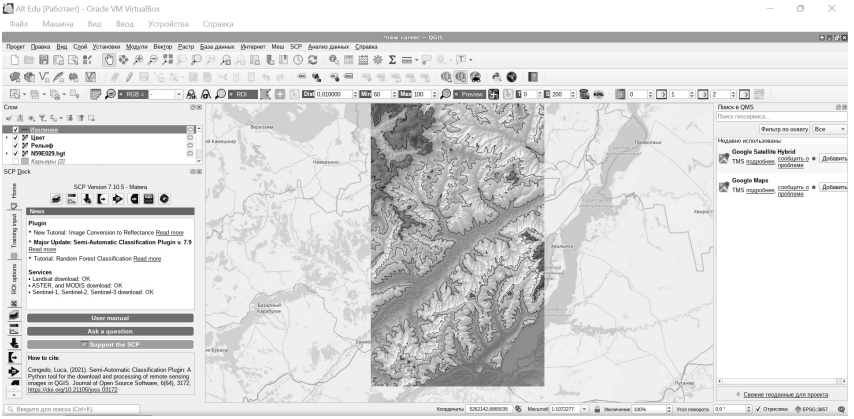


Рис. 1: Карта Вольского карьера

В операционной системе «Альт» доступна для установки программа Motion, которая поддерживает работу с видеоонлайн (рис. 3). Опишем преимущества данной системы.

- Поддержка работы с несколькими камерами.
- Непрерывная трансляция изображения с веб-камеры с сохранением транслируемого потока на носитель. Поддерживаются различные форматы поступающего видеопотока.

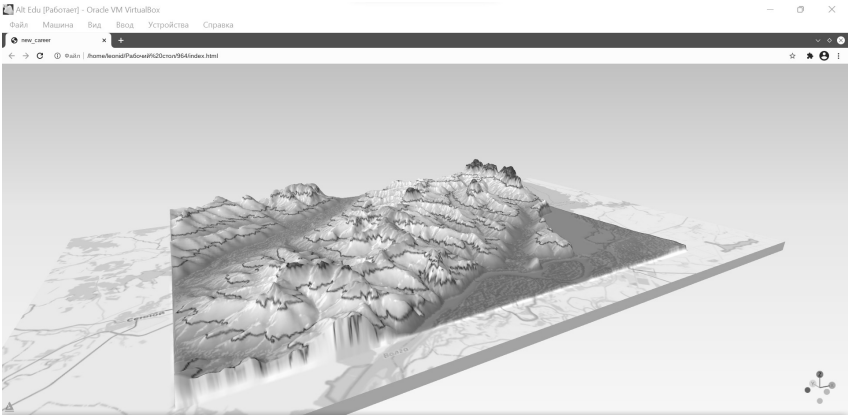


Рис. 2: Цифровая 3D-модель Вольского карьера

- Отслеживание наличия изменений на изображении, получаемом с камеры, и включение записи в случае обнаружения таких изменений — программа реагирует на движение и начинает запись только в этом случае.
- Трансляция изображения с камеры в сеть посредством собственного небольшого HTTP-сервера.
- Выполнение произвольного скрипта при заданном условии. Например, при обнаружении движения можно запускать скрипт оповещения по почте, смс, телефону или проигрывать трубный звук.

Проведём анализ преимуществ программного обеспечения, работающего на базе Alt.

Описание преимуществ. Прежде всего определяются основные аспекты, в которых и будут установлены преимущества использования ОС «Альт». Итак, достоинства могут быть классифицированы для Альт как для:

- Линуксоподобной ОС;
- Свободного ПО;
- Самостоятельной ОС.

Рис. 3: Скриншот видеопотока карьера Гатчинского района по производству щебня.

Альт — это целое семейство дистрибутивов Linux, которые унаследовали от него следующие достоинства: открытый исходный код ядра и основных компонентов распространяется абсолютно бесплатно, доступ к коду открывает огромные возможности для программистов и продвинутых пользователей, система нетребовательна к ресурсам, имеет удобную командную строку и продуманную файловую систему.

Свободное ПО позволяет использовать, адаптировать, распространять копии и публиковать свою улучшенную версию программы, что обеспечивается в том числе доступностью исходного кода и возможностью внесения в него модификаций и исправлений. При этом все производные должны распространяться под той же лицензией.

Как самостоятельная операционная система, Альт имеет крупное сообщество программистов — Alt Linux Team, большинство из которых не работает в «Базальт СПО», и собственный форум с активными пользователями. Первое даёт возможность любому программисту стать частью огромного продукта и поработать над его усовершен-

ствованием, а второе — получить оперативную помощь в затруднительной ситуации.

В Альт существует текущее стабильное ядро (std-def) и самое свежее ядро (un-def), поэтому при возникновении ошибок на новом, остаётся возможность использовать стабильное ядро, зачастую это оказывается очень полезной особенностью [7–10]. Кроме того, все пакеты репозитория «Сизиф» собираются с помощью среды hasher — собственной разработки Альт, обеспечивающей безопасную и воспроизводимую сборку благодаря тому, что не допускается влияние пакета на host-систему и взаимное влияние нескольких собирающихся пакетов друг на друга.

Выводы и заключение

Важным направлением реализации стратегии импортозамещения в области геоинформационных систем и технологий становится использование отечественных операционных систем для предобработки пространственных данных в управлении георисками. Особенно стоит выделить государственные организации, в которых уже существуют рекомендации, а в некоторых и требования, о полном замещении иностранных IT-продуктов. Что касается именно геоданных, например, съёмки с дрона могут захватывать важные стратегические объекты, тогда конфиденциальность и безопасность крайне важны для страны в целом, что и призваны обеспечить отечественные ОС.

Кроме того, дистрибутивы семейства Альт становятся популярными не только для использования определённых видов программного обеспечения, приложений и пакетов в управлении природопользованием на уровне федеральных структур Минприроды и Роснедра, а также региональных ведомств, но и самих предприятий горного сектора. Значит, всё более актуальным и важным является развитие инструментов анализа геоданных на базе указанных российских операционных систем.

Литература

[1] Vostrikov A. V., Prokofeva E. N., Goncharenko S. N., Gribanov I. V. Analytical modeling for the modern mining industry// Eurasian Mining. 2019. No.2(32). P.30–35. DOI 10.17580/em.2019.02.07

[2] Goncharenko S. N., Duong L. B., Petrov M. V., Stoyanova I. A. Modeling of parame-ters of innovation water-protection measures on the basis of industrial-technological indices of coal mining at Vietnam enterprises (2014) Gornyi Zhurnal, (9), pp.143–146.

[3] Prokofeva E. N., Vostrikov A. V., Shapovalenko G. N., Alvarez A. The development of effective geomonitoring for mining area with industrial review // Eurasian Mining. 2017. No. 2. P. 61–63.

[4] Temkin I., Deryabin S., Konov I.: Soft computing models in an intelligent open-pit mines transport control system. Procedia Computer Science, Vol. 120. 2017.

[5] Temkin, I.O., Klebanov D.A., Deryabin S.A., Konov, I.S.: Method of determining the state of the haul road career in the management of the interaction between robotic elements of the mining transportation complex. Mining journal, №1, P. 78–82. 2018.

[6] Rylnikova M., Ainbinder I., Radchenko D. Role of Safety Justification of Mining Development for the Regulatory Framework Formation and Mineral Resources Management 2018 E3S Web of Conferences 41, № 01033

[7] Курячий Г. В., Маслинский К. А. — Операционная система Linux — Национальный Открытый Университет «ИНТУИТ» — 2016 — 450с. — ISBN: 5-9556-0029-9 — Текст электронный // ЭБС ЛАНЬ — URL: https://e.lanbook.com/book/100278

[8] Современные операционные системы : 16+, Таненбаум Э., Бос, Х., ISBN: 978-5-4461-1155-8, 2019

[9] Курячий Г. В. — Операционная система Unix — Национальный Открытый Университет «ИНТУИТ» — 2016 — 258с. — ISBN: 5-9556-0019-1 — Текст электронный // ЭБС ЛАНЬ — URL: `https://e.lanbook.com/book/100281`

[10] Информационные ресурсы Alt ОС: `https://www.altlinux.orgPчhttps://www.basealt.ru`

А. Н. Непейвода
Переславль–Залесский, ИПС им. А.К.Айламазяна РАН
https://github.com/bmstu-iu9/TerminatorBattle2021

Эксперимент по созданию quick-and-dirty пруверов для оценки завершаемости в рамках рубежного контроля

Аннотация

Изложение теоретической информатики без возможности «потрогать руками» приводит к тому, что студенты начинают скучать, особенно те, кто уже умеют программировать и не считают, что теория способна улучшить этот навык. Однако привычные студентам задачи серьёзно отличаются от тех, которые связаны с практическим применением достаточно сложной теории. В частности, при разработке современных proof–assistant технологий или SMT-солверов учитывается изначальная неразрешимость задачи в общем виде и выбираются всевозможные аппроксимации её решения. В частности, такова задача проверки завершаемости систем переписывания термов[1].

Ежегодное соревнование пруверов по оценке завершаемости проводится с 2003 года [1]. Мы постарались сохранить в эксперименте его главные черты:

- никаких ограничений в языке реализации, при этом алгоритм (исходный код) решения должен быть открыт;
- система тестов также доступна участникам в процессе соревнования;
- никакого минимального набора тестов, которые прувер обязан точно проходить: подсчитывается лишь общее количество баллов за все тесты;
- наличие в базе данных очень сложных тестов, которые (почти точно) не будут решены ни одним прувером («монстров»);
- по желанию участники могут добавить к базе данных свои «секретные» тесты.

[1] С некоторыми оговорками можно было бы также вести речь о завершаемости программ на функциональном языке.

Было и два отличия. Во-первых, строить описание доказательства не требовалось, поскольку каждой команде пришлось бы ещё проектировать язык описания таких доказательств. Во-вторых, задача оценки завершаемости сложная и неоднозначная, а времени давалось слишком мало, чтобы успеть безупречно отладить пруверы, и дисквалификация за ошибки была бы чрезмерно жёстким решением в рамках рубежного контроля. В реальных состязаниях по завершаемости нет систем автоматической проверки генераторов доказательств, потому что нет навязанного единого способа строить такие доказательства, и бывали случаи, что правильному ответу соответствовало неправильное обоснование. Поэтому мы штрафовали неправильные ответы или формально правильные, но не обоснованные алгоритмом прувера[2] ответы только баллами.

Участие в эксперименте проводилось на добровольной основе: студенты могли сами выбрать, хотят они писать письменную контрольную работу или разрабатывать прувер. Накануне соревнования командам были выданы по 5 пробных задач из разных классов. На следующий день был выложен скрипт тестирования [2], оценивающий результат работы пруверов на 100 различных системах переписывания термов, часть из которых была сгенерирована автоматически, часть — взята из студенческих работ, а часть — из базы данных соревнований по завершаемости [3]. Код скрипта был полностью открыт, но написан на языке Рефал [4], и понять, как именно расшифровываются хеши тестов, было отдельной задачей (по желанию).

На «разборе полётов» выяснилось несколько выводов, которые студенты сделали при решении задачи.

1. На быстродействие при решении сложных задач больше влияет не язык реализации, а алгоритм. Участник, утверждающий, что проиграет по скорости, поскольку пишет на интерпретируемом языке Python, победил всех на самых коротких срезах, поскольку написал эффективный алгоритм предварительного анализа тестов.

2. Студенты сами отказались от идеи использовать нейросети для решения задачи (хотя это не запрещалось), поскольку минимальные изменения в синтаксисе теста могут привести к со-

[2] В частности, любые определённые ответы на вопрос о завершаемости систем–«монстров».

вершенно разным эффектам с точки зрения завершаемости, и нейросеть будет слишком часто ошибаться.

3. Правильный ответ не означает правильности программы. Если прувер формально корректно оценивал свойства «монстров», это позволяло сразу же построить тест, на котором выявлялось ошибочное поведение. Оказалось, что самый надёжный способ не получать штрафы — не пытаться ничего подгонять, а честно признаться, что решение построить не удалось.

В целом и по отзывам студентов опыт «игрушечной» научной разработки оказался положительным. Студенты убедились, что неразрешимые задачи решать можно и нужно, но без теоретической подготовки это не получится. Эксперимент также показал значительный разрыв в подготовке между студентами: несколько команд-добровольцев не смогли построить даже прототип прувера. Поэтому повторять такой опыт можно лишь будучи уверенными в том, что его участники хорошо владеют хотя бы базовыми алгоритмами синтаксического анализа и анализа графовых структур.

Литература

[1] *C. Marché, H. Zantema* The Termination Competition // Term Rewriting and Applications, 18th International Conference, RTA 2007, Paris, France, June 26–28, 2007, Proceedings, LNCS, 4533, pp. 303–313, 2007.

[2] *Ссылка на репозиторий проекта* `https://github.com/bmstu-iu9/TerminatorBattle2021`

[3] *Termination Problem Database* `http://www.lri.fr/~marche/tpdb/`

[4] *В. Ф. Турчин* РЕФАЛ-5: Руководство по программированию и справочник `http://www.refal.ru/rf5_frm.htm`

Лапшина Екатерина Александровна, Симонов Владимир Львович
Москва, Российский государственный социальный университет

Тенденции разработки программного обеспечения с использованием Low-code платформ

Аннотация

Рассмотрены варианты разработки программного кода с использованием свободного программного обеспечения, которое позволяет создать программный продукт с минимальным набором навыков через визуальные интерфейсы и конструкторы с помощью Low-code платформы.

Low-code платформы разработки — это приложение, которое предоставляет графический пользовательский интерфейс для программирования и, таким образом, разрабатывает код с большей скоростью и сокращает затраченные усилия с минимальным количеством кодирования. Подобные платформы реализованы в том числе, как свободное программное обеспечение (СПО) [1].

Такие среды разработки применяются для создания прикладного программного обеспечения через графический интерфейс пользователя вместо стандартного программирования вручную. С помощью платформ Low-code возможно создание полностью рабочего приложения, а в редких случаях — с использованием дополнительного кодирования. Данные среды разработки также помогают сократить объём программирования, что позволяет ускорить создание приложений. Большим преимуществом является то, что расширяется круг людей, которые могут внести свой вклад в разработку приложения. Low-code платформы также могут снизить первоначальные затраты на настройку, обучение и обслуживание.

Недавнее исследование бостонской компании Mendix показало, что спрос на разработчиков среди ИТ-специалистов достиг апогея. Почти шесть из десяти (57%) говорят, что количество персонала, необходимого для разработки программного обеспечения, увеличивается, а стоимость разработки программного обеспечения растёт (61%). [2] Также отметим, что в связи с растущими ожиданиями клиентов и изменением потребностей рынка после пандемии предприятия в разных отраслях всё больше проявляют инициативу в создании цифрового

контента для потребителей. Поэтому сегодня одним из актуальных решений является работа с Low-code платформами.

Рассмотрим варианты Low-code платформ для различных целей.

- NL!A framework — российский бесплатный low-code framework, позволяющий создавать полноценные бизнес-приложения. Модели, заложенные в кодогенератор NL!A framework, позволяют за считаные секунды создать полноценное рабочее бизнес-приложение [3];
- OutSystem — это надёжная и гибкая low-code платформа для разработки корпоративных мобильных и веб-приложений, которые разворачиваются в локальной или в гибридных средах [4];
- Mendix — это бескодовая (no-code) программная платформа, предоставляющая инструменты для создания, тестирования, развёртывания и проверки программных приложений [5].

Помимо бесплатных версий популярных платформ, также существуют различные варианты Low-code СПО с открытым исходным кодом, таких как Appsemble, Skyve, Baserow и другие. Наличие таких разнообразных платформ говорит о популярности данных решений и их развитии.

Есть ряд других причин, по которым предприятиям следует рассматривать Low-code платформы:

1. Более быстрый выход на рынок;
2. Повышение удовлетворённости клиентов;
3. Снижение затрат на ИТ-инфраструктуру;
4. Более эффективное управление приложениями;
5. Лучшее управление ИТ;

Безусловно, подобные среды разработки — это не панацея, а лишь вариант решения для создания программного продукта, поэтому Low-code платформы имеют ряд минусов:

1. Сложность в выборе подходящего ресурса;
2. Ограничения в функционале;
3. Зависимость от платформы;

Рассматривая плюсы и минусы Low-code платформ, можно сделать вывод, что они отлично подходят для быстрого создания небольших проектов и увеличения количества реализуемых решений. Они

сокращают разрыв между пользователями и разработчиками, что позволяет в короткий срок получить работающий прототип и сформировать видение будущей системы. Также можно проследить стремительное развитие данного направления в целом и как СПО.

Литература

[1] Платформа разработки low-code определение. — URL: https://www.hmong.press/wiki/Low-code_development_platform дата обращения 20.04.2022 г.

[2] David Roe, What's Behind the Explosion of Low-Code and No-Code Applications. — URL: https://www.reworked.co/information-management/whats-behind-the-explosion-of-low-code-and-no-code-applications/ дата обращения 20.04.2022 г.

[3] Программное обеспечение NL!A Framework . — URL: https://nl-a.ru/nla-framework дата обращения 20.04.2022 г.

[4] Программное обеспечение OutSystem. — URL: https://www.outsystems.com/ дата обращения 20.04.2022 г.

[5] Программное обеспечение Mendix. — URL: https://soware.ru/products/mendix дата обращения 20.04.2022 г.

[6] Лапшина Е.А., Симонов В.Л. Преимущества информационных систем с веб-интерфейсом // XIX Международная конференция «Современные информационные технологии в образовании, науке и промышленности» 29–30 апреля 2021 года, г. Москва.

Киселёв Денис Артурович, Симонов Владимир Львович
Москва, Федеральное государственное бюджетное образовательное учреждение высшего образования Московский авиационный институт (Национальный исследовательский университет)

Проект: Стенд динамических испытаний ЛА https://clck.ru/geKYo

Использование свободного программного обеспечения для моделирования стенда динамических испытаний элементов конструкций летательных аппаратов

Аннотация

Исследования динамического поведения элементов конструкции летательного аппарата при вибрационных испытаниях является неотъемлемым этапом производства. Причина состоит в том, что вибрационные, ударные и иные эксплуатационные перегрузки вызывают повреждения аппаратуры и конструкции, что часто приводит к неработоспособности изделия. Необходимость в механических испытаниях привела к созданию устройств формирования соответствующих воздействий, которые удобно реализовать программным способом. В работе приводится вариант реализации стенда динамических испытаний и моделирование его работы. Представлено программно-аппаратное средство для вырабатывания вибрационных и ударных сигналов по нескольким законам. Моделирование реализовано с использованием бесплатной онлайн-САПР TinkerCAD. Для программирования используется свободное программное обеспечение. Сделаны выводы о возможности применения представленной модели к реальным испытаниям.

Введение

Конструкция летательного аппарата (ЛА) подвергается вибрационным и ударным воздействиям на протяжении всего периода эксплуатации от воздействия аэродинамических эффектов, силовой установки и пр.

Вибрации вызывают накопление усталостных повреждений в конструкции ЛА, ухудшая качество всех видов механических соединений. Особенно опасны вибрации с частотами, совпадающими с собственными частотами элементов конструкции, т. к. возникающие резонансные колебания элементов конструкции могут приводить к опасным последствиям, вплоть до аварий.

Для обеспечения требуемой надёжности и ресурса элементов и конструкции ЛА в целом производитель должен обладать информацией об устойчивости изделий к вибрационным воздействиям. Важнейшую роль здесь играют механические испытания, регламентирующиеся рядом ГОСТ, основополагающими из которых являются ГОСТ: 24812-81; 59005-2020; 51371-99; 24346-80; 30630.0.0-99; 30631-99 [1] — [4] и ряд других.

Механические испытания проводятся с использованием специализированных установок и стендов. В настоящей работе рассматривает-

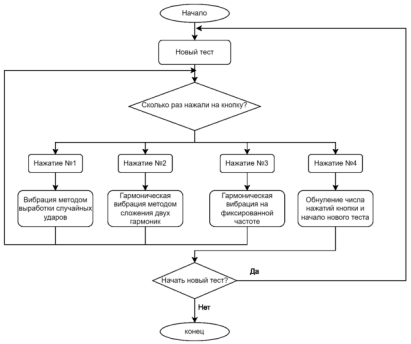


Рис. 1: Алгоритм работы модели

ся моделирование вибростенда для испытаний на линейную одноосевую вибрацию и удар.

На рисунке 1 представлен алгоритм работы стенда динамических испытаний, модель которого реализована в данной работе.

Задание формы воздействий осуществляется с помощью программных средств, относящихся к свободному программному обеспечению (СПО). Программно-аппаратная среда — Arduino IDE, электронные и электромеханические компоненты.

Данная разработка преследует в первую очередь учебные цели и не претендует на полноту охвата требований к промышленным испытательным системам [3].

В представленной разработке, при включении программы пользователю предлагается начать новое испытание изделия на вибрационные воздействия (три режима испытаний, соответствуют ГОСТ 24346-80 [4]):

- Случайная широкополосная вибрация;
- Полигармоническая вибрация (сложение двух синусоидальных сигналов);
- Гармоническая вибрация на фиксированной частоте.

Переключение режимов осуществляется последовательным нажатием тактовой кнопки. Последнее нажатие кнопки обнуляет число нажатий, и программа готова выполнять новый тест.

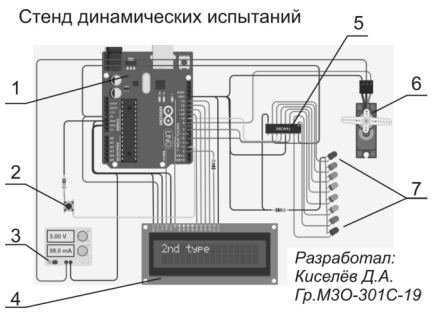


Рис. 2: Схема модели динамических испытаний

Реализация с использованием свободного программного обеспечения

Для разработки использована среда TinkerCAD [5]. Работа с СПО даёт возможность моделирования с использованием электронных компонентов (например, платы Arduino UNO) и создания программы на языке Си++ (версия Wiring).

Используемые компоненты: плата Arduino UNO; позиционный микросервопривод; 8-разрядный регистр сдвига; светодиоды разного цвета (жёлтые, зеленые, красные, оранжевые — по 2 шт.); жидкокристаллический экран; тактовая кнопка; резисторы; электрические провода, источник питания; вспомогательные компоненты, см. рис. 2.

Условные обозначения:
1 — плата Arduino UNO
2 — тактовая кнопка (управление режимами испытаний)
3 — источник питания микросервопривода
4 — жидкокристаллический экран
5 — восьмиразрядный сдвиговый регистр
6 — микросервопривод
7 — светодиоды индикации положения коромысла микросервопривода

Микросервопривод питается от отдельного источника. Коромысло микросервопривода совершает вибрационные движения и через соединительные механизмы присоединяется к вибростолу с объектом испытаний. Представленная схема с микросервоприводом сможет вос-

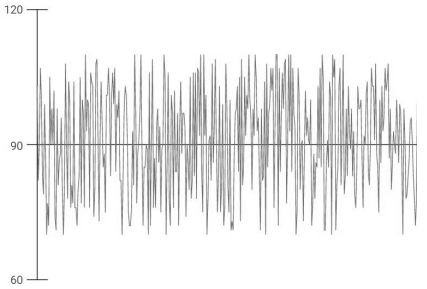


Рис. 3: Временная диаграмма случайной широкополосной вибрации

произвести колебания частотой не более нескольких герц (вследствие его инерционности), поэтому в реальных системах используют вибровозбудители электродинамического и иных типов.

Индикация углов поворота коромысла осуществляется светодиодными индикаторами. Начальное угловое положение коромысла — 90 градусов. В ходе работы положение коромысла меняется от 70 до 110 градусов (индицируется светодиодами определенного цвета). Например, при 105 — 110 градусах, загорается красный нижний светодиод. На ЖК-дисплее отображается информация о текущем режиме испытаний.

Результаты моделирования

При моделировании были получены следующие результаты. Для каждого режима построены временные диаграммы. Для первого режима испытаний (случайная широкополосная вибрация), временная диаграмма выходного сигнала представлена на рисунке 3. Для второго режима испытаний (гармонический сигнал, полученный сложением двух гармоник), временная диаграмма приведена на рисунке 4.

Для третьего режима испытаний — гармонический сигнал с фиксированной частотой и амплитудой — временная диаграмма приведена на рисунке 5.

Таким образом, можно провести испытания элементов конструкции летательного аппарата на вибрационные воздействия с последо-

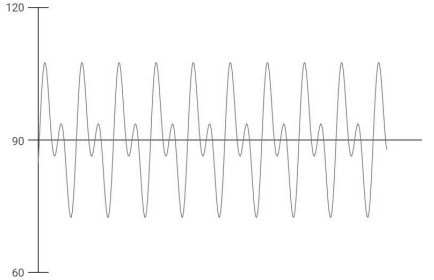


Рис. 4: Временная диаграмма полигармонической вибрации

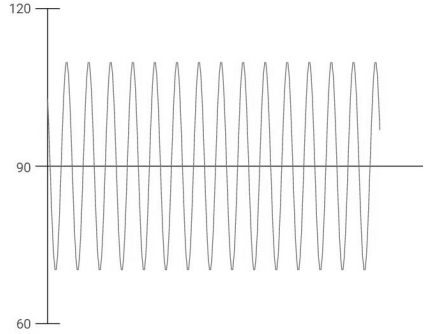


Рис. 5: Временная диаграмма гармонической вибрации

вательно устанавливаемыми режимами испытаний (временная диаграмма представлена на рисунке 7).

Конкретная циклограмма режимов испытаний задается разработчиками индивидуально для каждой конструкции.

Считывание данных о текущих реальных колебаниях элементов конструкции осуществляется вибродатчиками, разнообразными виброизмерительными приборами и системами, включая лазерные, оптические и т.д. (фирмы Zetlab, Микроникс, Brüel & Kjær, Ometron, Endevco и др.).

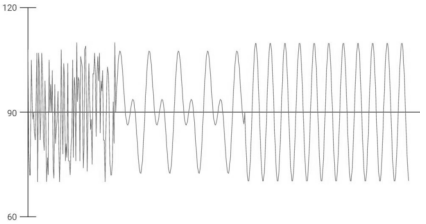


Рис. 6: Последовательная работа трёх режимов испытаний

Выводы

Разработано программно-аппаратное средство задания режимов проведения динамических испытаний элементов конструкции летательных аппаратов на вибрационные воздействия. Организовано выполнение трёх режимов испытаний (случайная широкополосная вибрация, гармоническая и полигармоническая вибрация), переключение между которыми осуществляет пользователь.

Данное устройство можно также использовать, например, для испытаний при производстве бортовых авиационных вычислителей и приборов. Такие объекты испытаний обладают относительно небольшими габаритами, что позволяет проводить испытания без потребности в высокой мощности исполнительных механизмов.

В качестве продолжения исследований планируется:

- расширение спектра возможных задающих воздействий (смесь случайной широкополосной и ударной вибрации; «качающаяся» гармоническая вибрация и т. д.), некоторые решения представлены в [6];
- сервомотор в настоящей разработке использован в качестве имитатора, поэтому предполагается замена на электродинамические системы.

Ссылка на проект (здесь же представлен программный код с использованием СПО): `https://clck.ru/geKYo`

Литература

[1] ГОСТ 24812-81. Испытания изделий на воздействие механических факторов. Общие положения. URL: `https://files.stroyinf.ru/Data/456/`

45681.pdf. Дата обращения 20.04.2022.

[2] ГОСТ Р 59005-2020 Авиационная техника. Комплекс лабораторный по исследованию прочности летательных аппаратов. Общие требования. URL: https://docs.cntd.ru/document/1200175704. Дата обращения 20.04.2022.

[3] ГОСТ Р 51371-99. Методы испытаний на стойкость к механическим внешним воздействующим факторам машин, приборов и других технических изделий. URL: https://docs.cntd.ru/document/1200010717. Дата обращения 20.04.2022.

[4] ГОСТ 24346-80. Межгосударственный стандарт. Вибрация. Термины и определения. URL: https://docs.cntd.ru/document/1200009496 . Дата обращения 20.04.2022.

[5] САПР онлайн-моделирования электронных схем TinkerCAD (фирма Autodesk). URL: https://www.tinkercad.com. Дата обращения 20.04.2022.

[6] Забегалова А. А., Новожилова А. Н., Симонов В. Л. (рук.). Программная реализация генератора шумов различной природы для проектирования информационных систем. В сборнике: Современные информационные технологии в образовании, науке и промышленности. XX Международная конференция. — Сб. трудов. Москва, 2021. С. 26–28.

Лапшина Екатерина Александровна, Рунов Александр Александрович, Ерпелев Алексей Владимирович, Каторгин Максим Константинович, Симонов Владимир Львович
Москва, Федеральное государственное бюджетное образовательное учреждение высшего образования «Российский государственный социальный университет» (ФГБОУ ВО РГСУ)
https://rgsu.net/

Использование свободного программного обеспечения при разработке устройств для развития мозговой и физической активности

Аннотация

Рассмотрена разработка устройств, компенсирующих статичную нагрузку на организм человека, возникающую при длительной работе за компьютерами и электронными устройствами, а также в условиях пандемии. Разработанные устройства содержат программно-аппаратные решения, использующие свободное программное обеспечение (СПО), позволяющие пользователю улучшить показатели физической и мозговой активности.

Введение

В условиях пандемии и последовавшей за ней самоизоляции много людей проводят время в статичной сидячей позе за компьютером, гаджетами, что ведёт к гиподинамии. В большой степени указанное относится к преподавателям, вынужденным, вместо аудиторных занятий с определённым уровнем двигательной активности, проводить время за компьютером вследствие дистанционного режима организации занятий. Студентов гиподинамия коснулась также в большой степени.

Гиподинамия привела к снижению двигательной активности и к её неизбежным последствиям: нарушениям опорно-двигательного аппарата, кровообращения, дыхания, пищеварения, сердечно-сосудистой системы и пр. От преимущественно сидячего образа жизни ухудшается обмен веществ, появляются застои в тазовых органах, увеличивается общий вес тела.

Гиподинамия отрицательно сказывается на работе головного мозга — появляется общая слабость, бессонница, чрезмерная утомляемость, уменьшаются трудоспособность, умственная активность.

Бороться с уже проявившимися и укоренившимися последствиями гиподинамии значительно труднее, чем их локализовать, купировать в ходе их развития. Поэтому в группе кружка программирования, электроники и робототехники в Российском государственном социальном университете была поставлена и решена задача разработки специализированных устройств для преодоления влияния гиподинамии и для тренировки мозговой и физической активности. За несколько лет было разработано более десяти таких устройств, и здесь представлены наиболее типичные.

Решение задачи разработки устройств компенсации последствий гиподинамии

Разработанные устройства относятся к программно-аппаратным средствам и основаны на микроконтроллерных и микропроцессорных платформах Raspberry Pi, Arduino [1], IskraJS, разнообразных электронных и электромеханических компонентах и т.д. Устройства размещались в специально разработанных корпусах, для чего применялось 3D-прототипирование и печать.

Для программирования применялось свободное программное обеспечение, например, для Arduino использовался язык C++ (версия Wiring).

Устройство 1: отслеживание осанки и времени работы за компьютером

Данное устройство вносит весомый вклад в решение поставленной задачи, в особенности для учащейся молодёжи [2]. Здесь контролируются несколько параметров: поза человека (а именно, сгорбился человек, или сидит прямо); допустимое для данной категории работников максимальное непрерывное время работы за компьютером. Имеется дополнительная функция сигнализации о недопустимой контрастности экрана компьютера по отношению к фону. Функционирование устройства состоит в следующем.

1. При превышении максимально допустимого времени пребывания за компьютером устройство сигнализирует и выдаёт настойчивые сигналы о необходимости сделать перерыв в занятиях.
2. Если осанка нарушена, прибор настойчиво сигнализирует об этом. Устройство выдаёт запрос на приведение позы в нормальное положение.
3. Если контрастность (соотношение яркости экрана по отношению к яркости фона) недопустима, прибор также настойчиво сигнализирует об этом. Автор данного проекта — Белан А. В., выпускник магистратуры Российского государственного социального университета, научный руководитель В. Л. Симонов.

Это устройство было протестировано в нескольких организациях, включая среднюю школу и университет. Были получены положительные результаты.

Устройство 2: биатлон в помещении

Этот достаточно перспективный проект позволяет в перерыве между занятиями выполнять активные двигательные действия, аналогичные соревнованию «Биатлон», а именно:

- провести серию выстрелов из лазерного пистолета;
- пройтись быстрым шагом или пробежать определённое расстояние;
- повторить серию выстрелов из лазерного пистолета и снова преодолеть дистанцию.

Таким образом, соревнуются несколько человек. При выполнении вышеперечисленных действий очки начисляются с учётом противоположных (по эффекту) действий: стрельбы из пистолета и перемещения (ходьбы или бега) по дистанции. В итоге побеждает тот, у кого твёрдая рука, и достаточно быстро передвигается. Студенты с удовольствием использовали данный проект.

Устройство 3: «Запомни и воспроизведи цветовую последовательность»

Целью третьего проекта «Запомни и воспроизведи цветовую последовательность» является создание игры для тренировки памяти на базе платформы для разработки электронных устройств Arduino Uno. Здесь тренируется память пользователя.

Сущность игры состоит в следующем. Электронное устройство вырабатывает цветовую последовательность (зажигание разноцветных светодиодов), которую следует запомнить и воспроизвести путём нажатия на соответствующие кнопки. С каждым шагом количество выработанных комбинаций увеличивается на одну. Для лиц с ограниченными возможностями здоровья по зрению предусмотрены звуки разных тонов. В результате сбора статистики было отмечено, что педагоги среднего и старшего возраста в среднем смогли воспроизвести последовательность из 25–30 шагов. Студенты воспроизводили иногда до 45–50 шагов.

Данная разработка является доступной для большого количества людей за счёт использования СПО и может применяться в образовательных целях. Все материалы, использованные при создании проекта, имеются у большинства не только разработчиков, но и людей

без специальной подготовки, а используемое программное обеспечение является бесплатным с открытым исходным кодом и не требует дополнительной установки, так как является информационной системой с веб-интерфейсом [3]. Данный проект разработан для условий пандемии, когда физическая и умственная активность людей снижена [4].

Отметим, что представленные разработки преследует в первую очередь учебные цели, в особенности, включая обучение лиц с ограниченными возможностями здоровья [5], и не претендует на полноту соответствия требованиям к серийно выпускаемым изделиям.

Выводы

Разработаны устройства, компенсирующие статичную нагрузку на организм человека, гиподинамию, возникающую при длительной работе за компьютерами, электронными устройствами и в условиях пандемии. Разработанные устройства содержат программно-аппаратные решения, использующие свободное программное обеспечение (СПО), позволяющие пользователям улучшить показатели физической и мозговой активности, о чём свидетельствуют замечания и отзывы всех, кто воспользовался разработками.

Ссылка на проект «Запомни и воспроизведи цветовую последовательность» (приведён фрагмент программного кода): `https://clck.ru/gf4s2`

Литература

[1] Сайт, посвящённый разработке крупных проектов на платформе Arduino. — URL: `https://alexgyver.ru/lessons/big-projects/`. Дата обращения 20.04.2022.

[2] Проблема гиподинамии студенческой молодёжи // Современные проблемы формирования здорового образа жизни у студенческой молодёжи: материалы Международной научно-практической интернет-конференции, 16–17 мая 2018 г., Минск, Беларусь / БГУ, Фак. социокультурных коммуникаций, Каф. экологии человека; редкол.: И. В. Пантюк (отв. ред.) [и др.]. — Минск: БГУ, 2018 г. — С. 234–237.

[3] Лапшина Е. А., Симонов В. Л. Преимущества информационных систем с веб-интерфейсом // XIX Международная конференция «Современные информационные технологии в образовании, науке и промышлен-

ности» 29–30 апреля 2021 года, г. Москва. — Стр. 62–65. URL:https://www.mairo.ru/files/archive/Sbornik19konferencii.2021.pdf. Дата обращения 20.04.2022 г.

[4] Лапшина Е. А., Симонов В. Л. Создание игры для тренировки памяти на базе платформы для разработки электронных устройств Arduino Uno // XIX Международная конференция «Современные информационные технологии в образовании, науке и промышленности» 29-30 апреля 2021 года, г. Москва. — Стр. 121–124. URL: https://www.mairo.ru/files/archive/Sbornik19konferencii.2021.pdf. Дата обращения 20.04.2022 г.

[5] Программирование с использованием СПО, электроники и робототехники как средство развития координации, моторики и реакции для лиц с заболеванием «Детский церебральный паралич» / Каторгин М., Ерпелев А., Селютин Д., Симонов В. В книге: СПО: от обучения до разработки. Объединённая конференция: сборник тезисов конференции. Москва, 2021. с. 116–119.

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

Применение свободного ПО и интерактивных элементов при изучении эргономики человеко-машинного взаимодействия

Аннотация

Приводится опыт преподавания эргономики интерфейса «человек-машина» для двух ступеней высшего образования специальностей профиля информатики и радиоэлектроники. Рассмотрено применение интерактивных элементов и передача пользовательского опыта (UX) за счёт средств виртуализации, распространяемых под свободными лицензиями, и элементов экспериментальных биометрических исследований.

Изучение дисциплин из цикла «Эргономика человеко-машинного взаимодействия» позволяет студентам специальностей профиля информатики и радиоэлектроники получить базовые знания в области инженерной психологии и эргономики, необходимые, чтобы решения, принятые при проектировании и последующем совершенствовании аппаратных и программных средств, обеспечивали эффективную работу человека-оператора.

Сложность процесса заключается в том, что предмет допускает только частичную формализацию. Существует множество факторов, которые формируют дизайн, и для грамотного прототипирования программных и аппаратных продуктов требуется большое количество визуального и субъективного практического опыта. Концентрированным выражением такого опыта может быть, в числе прочего, представление динамики исторического развития интерфейсов человеко-машинного взаимодействия [1].

Применительно к рассматриваемому циклу дисциплин нами использован ряд средств, позволяющих реализовать элементы интерактивного обучения.

Виртуализованная экспозиция по истории эволюции графических интерфейсов представляет собой набор виртуальных машин с исторически-значимыми операционными системами и прикладным программным обеспечением. Набор виртуальных машин является развитием концепции, представленной в [2]. В качестве системы виртуализации используется QEMU фиксированных версий, собираемая из исходных кодов (для исключения возможных проблем совместимости с виртуализованными объектами после апгрейда системы). Дисковые образы используются в режиме copy-on-write, с использованием неизменяемого базового и изменяемых производных образов, а само практическое задание содержится в мгновенном снимке (снапшоте) виртуальной машины с предварительно запущенным и готовым к немедленному использованию ПО. Для запуска ряда экспонатов применяются эмуляторы, работающие внутри QEMU в режиме вложенной виртуализации.

Экспозиция включает разделы по истории настольных и мобильных графических операционных систем (1, 2), раздел по эволюции графических оболочек GNU/Linux (3), а также разделы по эволюции текстовых процессоров (4) и, фрагментарно, графических редакторов (5). Экспонаты в некоторых разделах полностью являются свободным ПО, как в случае (3), либо смесью свободного ПО и abandonware. При формировании заданий отдаётся предпочтение первой категории.

Практические задания сводятся к выполнению однотипного набора действий последовательно в трёх разных программных системах, отражающих этапы эволюции UX и построенных с использованием различающихся метафор и визуальных средств, с последующим выполнением usability-анализа.

- Хорошо зарекомендовавшим себя вариантом задания является ручная сортировка файлов (обеспечивается практическое сравнение двухпанельных, навигационных и пространственных моделей файловых менеджеров, а также оценивается физическая нагрузка при использовании перетаскивания объектов и без него). В качестве образцов использованы экспонаты с оболочкой GNOME из раздела (3).
- Рисование в растровых графических редакторах позволяет на практике оценить влияние принципа прямого управления на интуитивность интерфейса. В качестве редактора, не использующего данный принцип, используется Xerox Alto Draw, предзапущенный в эмуляторе SALTO [3] (с обучающим видео, без которого невозможно использование редактора), а его применение изучается на основе Adobe Photoshop 1.0.1 (версия, исходные коды которой выложены в открытый доступ Музеем компьютерной истории) в эмуляторе Mini vMac [4] и любом из Paint-подобных редакторов Linux.
- Ознакомление с эволюцией принципа WYSIWYG при работе с текстовыми документами выполняется на основе EZ Word из состава Andrew User Interface System, а также текстовых процессоров Ted и Abiword (все три виртуализованы на базе дистрибутива Debian).

Также в практическую часть входит **получение практического опыта использования манипуляторов** различной конструкции в процессе человеко-машинного взаимодействия и его анализ на базе ряда действующих музейных манипуляторов (средств управления сценой и курсором). Экспозиция по эволюции средств управления курсором является ещё одной собственной разработкой, и представляет из себя набор стендов с образцами манипуляторов типа «мышь», «трекбол» и «спейсбол», выпускавшихся с 1980-х годов. При знакомстве с ними обучаемые получают непосредственное представление об этапах эволюции формы манипулятора, влиянии на неё использовавшихся физических принципов и изменений в парадигмах пользовательского интерфейса.

Дополнительным средством, обеспечивающим приобретение студентами личного исследовательского опыта по профилю изучаемых дисциплин, является **участие в реальных экспериментах по биометрическому мониторингу** взаимодействия человека-оператора

с программными и аппаратными продуктами. В ходе сравнительной оценки интерфейса «человек-машина» обучаемые принимают участие в натурном исследовании, включающем получение и анализ показателей скорости работы (темп выполнения операций, число ошибок) и биометрических данных, характеризующих испытываемые пользователем нагрузки (пульс, ритмы головного мозга, направление взгляда) [5]. Особенность проводимых натурных экспериментов — совмещение самосообщаемых параметров, оцениваемых по классическим методикам, с анализом данных, получаемых от биометрических устройств бытового сегмента (фитнес-трекеров, энцфефалографов, айтрекеров). Участвуя в натурных экспериментах, студенты приобретают ряд академических и профессиональных компетенций, включая навыки планирования и проведения испытаний и исследований, системного и сравнительного анализа, анализа эксплуатационных свойств объектов, оценки конкурентоспособности изделий и выработки требований к их модификации.

По результатам работы проводится круглый стол, где студенты могут поделиться своим видением изучаемой проблематики.

Использование комплексного подхода, сочетающего традиционные и интерактивные методы обучения, помогает более качественно сформировать будущих специалистов за счёт передачи дизайна через субъективный опыт и формирования ментальных моделей и объективных представлений о действительности, тогда как использование только традиционного подхода оставляет после себя лишь абстрактные представления об эргономике. За счёт демонстрации большого количества исторически-значимых интерфейсов и устройств, обеспечивающих человеко-машинное взаимодействие, увеличивается визуальный субъективно-практический опыт. Участие в натурном эксперименте позволяет студентам приобрести профессиональный опыт организации исследований, подход к системному многофакторному анализу, что значимо для формирования будущего специалиста. Организация круглого стола, в свою очередь, обеспечивает приобретение ораторских навыков, повышает у студентов их уровень критики и самокритики. В целом использование данного подхода обеспечивает развитие не только перечисленных компетенций студентов, но и в целом использование знания основ социологии, физиологии и психологии труда.

Литература

[1] *Волчецкий И., Костюк Д., Луцюк П., Сойко Ю.* Построение документации с живыми иллюстрациями на основе встроенных виртуальных машин // Шестнадцатая конференция разработчиков свободных программ: тез. докл. / Калуга, 27-29 сентября 2019 г. — М.: Базальт СПО, 2019. — С. 59-64.

[2] *Костюк Д.А.* Особенности использования виртуализованных окружений, внедрённых в презентационные материалы // Восьмая конференция «Свободное программное обеспечение высшей школе»: тез. докл. / Переславль, 26–27 января 2013 года. М.: Альт Линукс, 2013. — С. 83–86.

[3] SALTO - Xerox Alto I/II Simulator `https://github.com/brainsqueezer/salto_simulator`

[4] Mini vMac - early Macintosh emulator `https://www.gryphel.com/c/minivmac/index.html`

[5] *Костюк Д.А., Маркина А.А.* Подход к комплексному межгрупповому usability-тестированию для платформы GNU/Linux // Тринадцатая конференция «Свободное программное обеспечение в высшей школе»: Материалы конференции. — Переславль, 26–28 января 2018 г. — М.: Basealt, 2018. — С. 39–44.

Воронин И.В., Дарсавелидзе А.А.
Шатура, Московской области, ИПЛИТ РАН

Развёртывание нейросети на базе ОС «Альт» для обнаружения онкологических заболеваний

Аннотация

В статье обсуждаются способы развёртывания и обучения глубокой сверточной нейросети в контейнере Docker, на базе ОС Альт. Рассмотрены необходимые ресурсы для создания разных моделей распознавания результатов анализов пациентов по спектрограммам. Предлагается решение — для определения онкологических заболеваний по спектрограммам из карт опухолевой области мозга.

В современной медицине активно развиваются новые решения в области обработки и анализа данных полученных при помощи рамановской спектроскопии или спектроскопии комбинационного рассеяния — когда спектроскопический метод исследования используется

для определения колебательных мод молекул и вибрационных мод в твёрдых телах. В данной работе проводится анализ спектрограмм, на основе которых можно диагностировать и различить больную ткань живого человека от здоровой. Для такой диагностики и распознавания спектров тканей была использована глубокая свёрточная нейросеть из пакета Keras — официального бэкэнда Tensorflow.

Оболочка Jupyter, делает использование Python намного проще и интуитивно понятнее даже для человека, далёкого от программирования. Существуют платные серверы, где можно развернуть и использовать данную среду, с автообновлением и регулярными backup-ами. В данной работе мы развернули нашу собственную нейросеть на серверном узле, с пропускной способностью сети гигабит в секунду:

`http://astera.laser.ru:8888/?token=c4d16a340eab7fbc5b285effd01127b0ada478413fb9b9ad`

В нашем случае мы использовали уже предустановленный Docker — сконфигурированный для развёртывания на множестве серверов.

```
$ docker-compose up -d
```

Определить адрес токена для доступа к серверу с запущенной нейросетью можно по команде:

```
$ docker logs tf_test
```

От медицинских работников были получены спектрограммы здоровых и больных тканей человека. Для обучения сети была обработана выборка порядка 1000 спектрограмм. Сеть развёртывалась в операционной системе — на российской платформе Alt p10. Основные вычисления производились на CPU сервере. Обязательным условием в нём должна быть инструкция AVX, наличие которой можно диагностировать следующей командой:

```
 $cat /proc/cpuinfo |grep avx
```

Каждый файл исходных данных содержит информацию о длине волны и интенсивности. Для разбора итоговых данных мы закодировали результаты в матрицу:

1,0,0 — abouttumoral (околоопухолевая область)

0,1,0 — healthy (здоровая область)

0,0,1 — sick (опухолевая область)

Делим датасет на тренировочную часть и тестовую в соотношении 85 к 15 параметром test_size=0.15

Были использованы предопределённые классы для слоёв:

- Dense() — полносвязный слой;
- Conv1D, Conv2D — свёрточные слои;
- MaxPooling2D, Dropout, BatchNormalization — вспомогательные слои

А также предопределённые классы моделей:

- Model — общий класс модели;
- Sequential — последовательная модель.

У каждого слоя и у модели в целом имеется свойство weights, содержащее список настраиваемых параметров (весовых коэффициентов). В нашем случае сеть в себя включает 16,757,443 параметров.

Создаём архитектуру модели, которая является основой для определения MNIST dataset:

```
model2 = Sequential()
model2.add(Conv1D(128, 4, activation='relu', input_shape=(1015,1),kernel_regularizer=
                                  regularizers.l1_l2(l1=1e-5,l2=1e-4))) # 32 neurons
model2.add(Conv1D(128, 4, activation='relu', bias_regularizer=regularizers.l2(1e-4)))
                                                                        # 32 neurons
model2.add(BatchNormalization())
model2.add(Activation('relu'))
model2.add(MaxPooling1D())
model2.add(Dropout(0.25))
model2.add(Conv1D(256, 2, activation='relu', kernel_regularizer=regularizers.l1_l2(
                                                  l1=1e-5, l2=1e-4))) # 64 neurons
model2.add(Conv1D(256, 2, activation='relu', bias_regularizer=regularizers.l2(1e-4)))
                                                                        # 64 neurons
model2.add(BatchNormalization())
model2.add(Activation('relu'))
model2.add(MaxPooling1D())
model2.add(Dropout(0.25))
model2.add(Flatten())
model2.add(Dense(256, activation = 'relu', use_bias=False))
model2.add(BatchNormalization())
model2.add(Activation('relu'))
model2.add(Dense(128, activation = 'relu', use_bias=False))
model2.add(BatchNormalization())
model2.add(Activation('relu'))
model2.add(Dense(64, activation = 'relu', use_bias=False))
model2.add(BatchNormalization())
model2.add(Activation('relu'))
model2.add(Dropout(0.25))
model2.add(Dense(3, activation = 'softmax'))
model2.summary()
model2.compile(loss='categorical_crossentropy',optimizer='adam',metrics=['accuracy'])
model2_hist = model2.fit(xtrain, ytrain ,batch_size=128, epochs=100, verbose=1)
```

```
Total params: 16,757,443
Trainable params: 16,755,779
Non-trainable params: 1,664
```

```
Epoch 1/100
1/1 [==============================] - 2s 2s/step - loss: 1.7923 - accuracy: 0.2321
Epoch 100/100
1/1 [==============================] - 1s 713ms/step - loss: 0.3576 - accuracy: 0.9107
```

Чтобы оценить итоговую точность модели на тестовой части датасета, выполняем следующие команды:

```
acc = model2.evaluate(xtest, ytest)
print("Loss:", acc[0], "Accuracy:", acc[1])
pred = model2.predict(xtest)
print(np.round(pred,2))
1/1 [==============================] - 0s 229ms/step - loss: 9.8597 - accuracy: 0.9636

[[0.09 0.91 0.  ]
 [0.01 0.99 0.  ]
 [0.  1.  0.  ]
 [0.  1.  0.  ]
 [0.1  0.9  0.  ]]
```

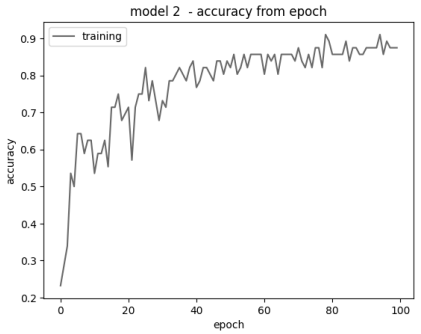


Рис. 1: model 2 — accuracy from epoch

Полученный результат говорит о том, что тестовые спектрограммы были распознаны с вероятностью 91% для здоровых тканей.

Литература

[1] https://keras.io/api/layers/

[2] https://keras.io/api/models/

[3] https://keras.io/guides/training_with_built_in_methods/

[4] https://proproprogs.ru/tensorflow/keras-posledovatelnaya-model-sequential

Зыкин Иван
Наро-Фоминск, ИАТЭ НИЯУ МИФИ, ООО «Процессные технологии»
Проект: RunaWFE Free https://runawfe.org/

Разработка элемента «бизнес-правило» для свободной системы RunaWFE Free в качестве производственной практики и ВКР

Аннотация

Бизнес-правила в системах управления бизнес-процессами используют для того, чтобы вынести из схемы бизнес-процесса в элемент «бизнес-правило» часть бизнес-логики и таким образом упростить схему бизнес-процесса, а также сделать её менее подверженной изменениям. В докладе рассказывается о реализации элемента «бизнес-правило» для свободной системы RunaWFE Free в рамках производственной практики и ВКР во время обучения автора в ИАТЭ НИЯУ МИФИ.

Система управления бизнес-процессами

Система управления бизнес-процессами — это программное обеспечение, предназначенное для сокращения времени выполнения процессов предприятия, за счёт их регламентации, автоматизации и прозрачности для всех участников. Использование таких систем эффективно для предприятий, в производственной деятельности которых происходит многократное повторение заранее известных цепочек действий, совершаемых различными исполнителями.

Одной из составляющих системы управления бизнес-процессами RunaWFE Free является среда разработки (Рис. 1), предназначенная

для моделирования бизнес-процессов в графической нотации BPMN (англ. Business Process Model and Notation, нотация и модель бизнес-процессов).

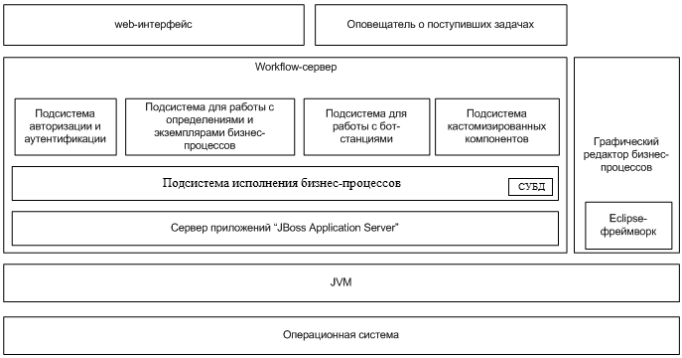


Рис. 1: Архитектура RunaWFE Free.

Среда разработки BPMS RunaWFE Free представляет собой RCP-приложение и технологически построен на основе GEF (Graphical Editing Framework).

Платформа Eclipse представляет собой набор подсистем, реализованных небольшим исполняемым приложением ядра и набором модулей (плагинов), расширяющих функциональность платформы.

Для расширения функциональности, плагины платформы в дескрипторах плагинов определяют точки расширения (extension points). Точка расширения представляет собой xml описание интерфейса расширяемого компонента плагина.

GEF предоставляет основу для создания графических редакторов. GEF реализована как набор плагинов расширяющих плагины платформы Eclipse. GEF связывает элементы модели приложения с их графическими представлениями, реализованными с помощью графических компонент библиотеки Draw2d. Контроллеры GEF поддерживают визуальное представление элементов модели в MVC (model-view-controller) архитектуре.

Таким образом, реализация элемента «бизнес-правило» заключается в добавлении нового элемента к существующей точке расширения elements и разработке java классов согласно MVC архитектуре.

Бизнес-правило

`Бизнес-правило` является элементом графической нотации BPMN (Рис. 2).

Функциональное предназначение элемента заключается в исполнении какой-либо формулы, при выполнении заданного условия.

Внедрение элемента `Бизнес-правило` в систему управления бизнес-процессами предоставляет следующие преимущества:

- сокращение времени разработки;
- быструю реакцию на изменения;
- упрощение общего дизайна системы;

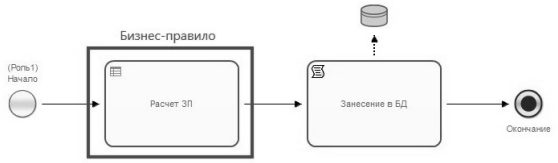


Рис. 2: Схема бизнес-процесса.

Реализация элемента «бизнес-правило» в свободной системе RunaWFE Free

В ходе работы были написаны следующие классы:

- BusinessRule — класс, непосредственно представляющий элемент (иконку) «Бизнес-правила» в среде разработки;
- BusinessRuleProvider — класс, инициализирующий создание диалогового окна, при возникновении событий нажатия на элемент и производящий валидацию конфигурации элемента.
- BusinessRuleEditorDialog — класс, создающий и отображающий окно редактирования конфигурации элемента, основанный на визуальных компонентах фреймворков SWT и JFace (Рис. 3);
- BusinessRuleModel — класс, сохраняющий в себе параметры конфигурации;

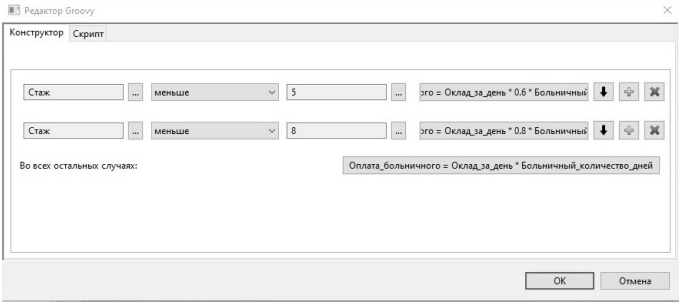


Рис. 3: Окно редактирования конфигурации элемента.

- BusinessRuleHandler – класс, отвечающий за обработку и выполнение скрипта на сервере.

Заключение

Разработанный код был загружен на портал разработчиков свободного программного обеспечения GitHub в репозиторий компании «Процессные Технологии» — `https://github.com/processtech`

На основании работ по этому направлению автор прошёл производственную практику в компании «Процессные Технологии» и защитили ВКР в ИАТЭ НИЯУ МИФИ.

Кулагин Владимир Петрович, д.т.н., профессор, Корепанов Вячеслав Дмитриевич, магистрант
Москва, Российский технологический университет — МИРЭА
`https://github.com/korepanov/bint`

Разработка интерпретатора с шифрованием для С-подобного языка

Аннотация

В настоящее время существуют различные алгоритмы шифрования данных, а также обфускации программного кода. Так, представляется возможным зашифровать исходный код программы, а затем

дешифровать его, собрать и исполнить. Однако в этом случае лицо, выполняющее программу, имеет доступ к её исходному тексту. Другим подходом является использование различных обфускаторов, однако обфусцированную программу может исполнить любое лицо, в том числе нежелательное. Возможно сначала обфусцировать программу, а затем зашифровать её, затруднив тем самым понимание программы лицом, собирающим и запускающим программу. Однако существуют определённые трудности такого подхода в связи с тем, что лицо, обладающее обфусцированным исходным кодом программы, должно также обладать соответствующим программным обеспечением по шифрованию и дешифрованию; компилятором, либо интерпретатором языка, а также иметь знания по использованию этого инструментария. В докладе рассматриваются вопросы конструирования и разработки интерпретатора с шифрованием, позволяющего исполнить программу только лицом, обладающим ключом, а также затруднить её понимание.

Общий алгоритм работы интерпретатора с шифрованием

Исходная программа пишется на собственном языке, называемом языком B. Синтаксис языка B определяется синтаксисом языка C, за исключением некоторых его возможностей. По этой причине мы не будем подробно останавливаться на синтаксисе разрабатываемого языка. Отметим только, что в настоящее время язык B является языком Basm, с возможностью использования C-подобных функций. Язык Basm — Assembler-подобный язык. Подробное описание синтаксиса языка Basm может быть найдено в файле README.md репозитория github[1].

Структурная схема интерпретатора с шифрованием представлена на рисунке 1.

Программа, написанная на языке B, подвергается трансляции на язык Basm с помощью соответствующих модулей, написанных на языках B и Basm. Модули, написанные на языке Basm, могут быть превращены в модули на языке Go с помощью специально написанного для этих целей транспилера, после чего можно получить исполняемые файлы этих модулей. Причина, по которой не происходит непосредственной интерпретации модулей на языке Basm, а собираются исполняемые файлы, — требование высокой производительности этих модулей. Модули, написанные на языке B, транслируются в язык Basm с использованием ранее сформированных исполняемых модулей. Таким образом, новые конструкции языка B пишутся на старых кон-

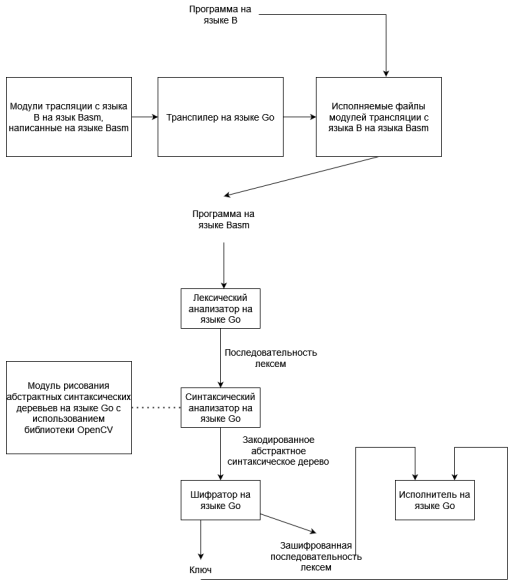


Рис. 1: Структурная схема интерпретатора с шифрованием

струкциях языка B. Тем самым происходит верификация правильности интерпретации конструкций языков B и Basm, что повышает надёжность интерпретатора. Транслированная программа с языка B на язык Basm подвергается лексическому и синтаксическому анализам, после чего происходит её шифрование с возможностью последующего исполнения лицом, обладающим ключом. С целью облегчения разработки написан необязательный модуль рисования абстрактных синтаксических деревьев (АСД) с использованием библиотеки OpenCV. Модуль считается «необязательным» по той причине, что он не участвует в сборке окончательной версии продукта и потому связь с этим модулем на рисунке 1 показана пунктирной линией.

Лексический анализ

Лексический анализ заключается в разбиении исходного множества символов, пришедшего на вход, на элементарные логические единицы — лексемы [3, 4].

Определены следующие типы лексем[2]:

VAR	Переменная
OP	Операция
BR	Круглая скобка
CD_BR	Квадратная скобка
VAL	Значение
SEP	Разделитель

Таблица 1: Типы лексем языка Basm

Синтаксический анализ

На предыдущей стадии мы разбили команды на лексемы. Однако мы не знаем, в какой именно взаимосвязи эти лексемы находятся друг по отношению к другу. Связи лексем, как и сами лексемы, определяются синтаксисом языка. Задача синтаксического анализа — установить эту взаимосвязь. Для этого применяется аппарат АСД[2].

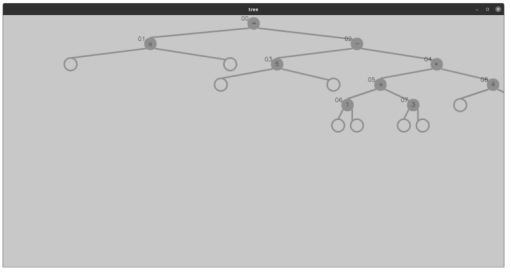


Рис. 2: Пример АСД для команды «a = (5 — ((1 + 3) * 4))»;

Результат закодированного АСД для команды «a = (5 — ((1 + 3) * 4));»:

=	a	-	5	*	+	1	3	4

Шифрование и дешифрование

Шифрование результата, закодированного АСД, заключается в добавлении мусора (произвольных результатов, закодированных АСД, для реально существующих команд языка Basm) в позиции между ячейками и запоминании смещений для реальных, не являющихся мусорными, ячеек относительно начала в отдельный файл. Данный файл является секретным ключом. Далее программа может быть выполнена лицом, обладающим ключом, когда оно подаст на вход исполнителю зашифрованную программу и ключ.

Литература

[1] https://github.com/korepanov/bint/blob/master/README.md

[2] https://github.com/korepanov/bint/blob/master/docs/P§P«CSCЃP,,РчСЂРхРьСЖР,,РѳCSPчРѳР»РчСЃРэРѳ.odt

[3] *Альфред В. Ахо, Моника С. Лам, Равн Сети, Джеффри Д. Ульман.* Компиляторы: принципы, технологии и инструментарий = Compilers: Principles, Techniques, and Tools. — 2-е изд. — М.: Вильямс, 2008, 1184 с.

[4] *Робин Хантер.* Основные концепции компиляторов = The Essence of Compilers. — М.: «Вильямс», 2002, 256 с.

Валерий Баканов
Москва, МИРЭА (Российский технологический университет РТУ МИРЭА), НИУ ВШЭ (Национальный исследовательский университет ВЫСШАЯ ШКОЛА ЭКОНОМИКИ).

Проект: Data-Flow, SPF@home http://vbakanov.ru/dataflow/, http://vbakanov.ru/spf@home/

Параллелизм в алгоритмах — выявление и рациональное использование

Аннотация

В современных технологиях обработки информации одной из важнейших является параллелизация, обеспечивающая повышение производительности и надёжности обработки данных. В то же время преподавание соответствующих дисциплин зачастую поставлено излишне формально и не полностью соответствует сегодняшним технологическим реалиям в данной области знаний и навыков. В связи с разработкой новых (отечественных) микропроцессоров важно получение студентами компетенций в области реализации параллельных вычислений, максимально использующих архитектурные особенности разрабатываемых аппаратных средств.

Автор уверен в верховенстве сущности «алгоритм» при глубинном рассмотрении проблемы параллелизма, которая должна предшествовать механическому программированию с опором на библиотечные программы. При такой постановке задачи возрастает ценность изучения внутренних свойств именно алгоритмов как реальных блоков построения программ [1]. Важнейшим этапом является анализ тонкой информационной структуры конкретных алгоритмов (программ) [2].

В данной работе исследуемый алгоритм представляется в форме операторов императивного ассемблероподобного языка, причём никакой априорной информации о потенциале скрытого параллелизма не имеется. Конечной целью является создание плана (расписания) параллельного выполнения заданного алгоритма на определённом (возможно, гетерогенном) поле параллельных вычислителей (далее ПВ). Такая концепция в целом соответствует подходу ILP (*Instruction-Level Parallelizm*, параллелизм уровня машинных инструкций).

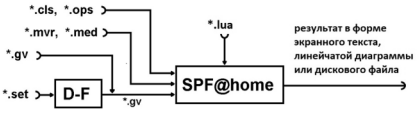


Рис. 1: Общая схема инструментального программного комплекса для построения планов выполнения параллельных программ (*.set и др. – типы файлов, с которыми работает программный комплекс)

На вход программной системы (рис. 1) поступает описание анализируемого алгоритма в указанной форме или формального его описания в виде ориентированного ациклического Информационно-

го Графа Алгоритма (далее — ИГА) — зависимость вида "операторы → операнды", при этом вершины графа ассоциируются с операторами (группами операторов) программы, а дуги — с линиями передачи данных.

Выявление и анализ внутреннего логического параллелизма в алгоритмах реализовано с помощью имитации модели акторов (модуль D-F) и построения специальных сечений ИГА в виде его Ярусно-Параллельной Формы (далее — ЯПФ, модуль SPF@home). Оба упомянутых модуля разработаны с использованием языка C/C++ в стиле GUI для модели Win'32, являются полностью OpenSource[1] и могут быть выгружены для свободного использования (формат инсталляционных файлов)[2]; рис. 2.

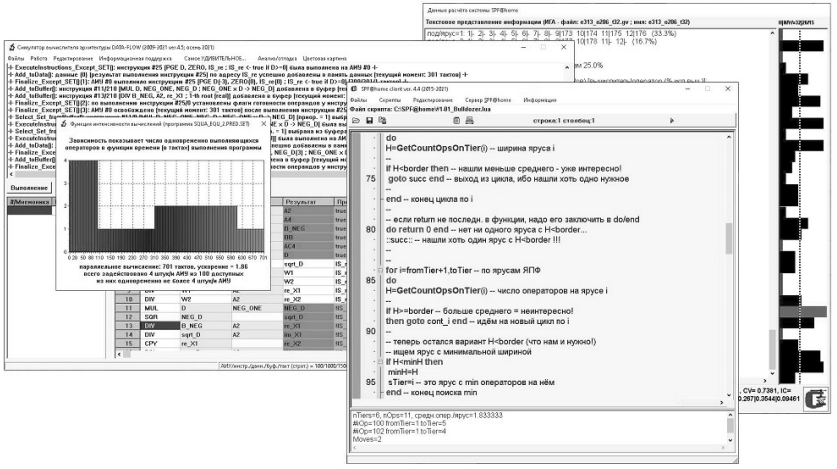


Рис. 2: Пользовательский интерфейс в стиле GUI программ рассматриваемого комплекса СПО

Модуль D-F (Data-Flow) является фактически универсальным вычислителем архитектуры SMP (*Symmetric MultiProcessing*, системы с

[1] https://github.com/Valery-Bakanov

[2] http://vbakanov.ru/dataflow/content/install_df.exe, http://vbakanov.ru/spf@home/content/install_spf.exe

общей памятью), на вход которого подаётся последовательная программа (используются 3-х символьные мнемоники команд и 3-х адресная система с порядком следования операндов согласно соглашениям AT&T, [3]). Условное выполнение реализуется методом предикатов, для реализации циклов используется система макросов, "разворачивающих" циклические структуры. Удобству визуализации решения способствует выдача данных о процессе выполнения программ в виде функции интенсивности вычислений (число одновременно выполнившихся операторов в функции времени) и диаграмм Ганта.

Модуль SPF@home предназначен для моделирования и выбора наилучших (в заданном смысле) сценариев преобразования ЯПФ [4] как планов параллельного выполнения программ на вычислительной системе определённой архитектуры [5]. Программный модуль SPF@home строит план выполнения параллельной программы без привязки к конкретной технологии параллельного программирования, поэтому логичнее говорить о *каркасе* плана (расписания) выполнения (рис.3).

Задача построения *каркаса* плана выполнения параллельной программы в общем случае является *NP*-полной [6], поэтому в данном случае используется эвристический подход к решению. Собственно сценарий преобразования реализуется с использованием скриптового языка Lua [7], при этом Lua-вызовы являются «обёртками» над функциями API системы SPF@home.

Конечной целью применения сценариев целенаправленного преобразования ЯПФ является фактически список EPIC (*Explicitly Parallel Instruction Computing*, набор инструкций с явным параллелизмом) для выполнения на конкретной параллельной вычислительной системе (особо подходящей архитектурой можно считать VLIW (*Very Long Instruction Word*, сверхдлинное машинное слово).

Типовые студенческие задания исследовательского уровня, при решении которых используются программные модули `D-F` и `SPF@home`:

- Определить число ПВ, при которых время параллельного выполнения заданного алгоритма минимально. Насколько увеличится время при половинном (30% ... 70%) числе ПВ?

- Каким образом минимальное число ПВ зависит от размерности обрабатываемых данных (при неизменном алгоритме их обработки)?

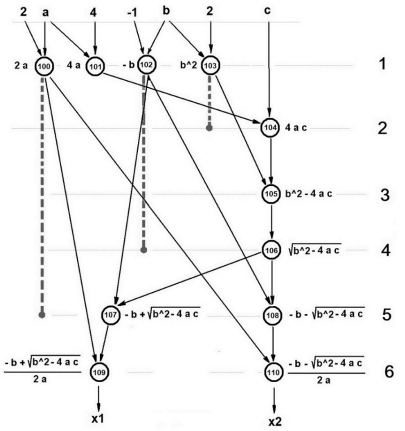


Рис. 3: ЯПФ алгоритма как *каркас* плана выполнения параллельной программы нахождения действительных корней полного квадратного уравнения в «верхнем» варианте (номера ярусов приведены справа, пунктиром показаны допустимые положения операторов по ярусам ЯПФ

- Сравнить два (несколько) заранее разработанных сценариев целенаправленного преобразования ЯПФ для одного (нескольких) алгоритмов для выполнения на заданном поле ПВ?
- Разработать сценарий целенаправленного преобразования ЯПФ для выполнения заданного алгоритма (алгоритмов, классов алгоритмов) на заданном поле ПВ исходя из заданных требований (минимума вычислительной сложности процедуры преобразования, максимума равномерности загрузки ПВ, минимума времени выполнения алгоритма).

Следует заметить, что модуль SPF@home обладает большими возможностями, чем декларировано идеологемой ILP. Рассматривая сущность «операторы» как группы операторов любого размера, этот модуль полезен и при составлении планов ПВ программ с гранулами параллелизма большого размера — напр., в МИРЭА эта возможность используется при занятиях на кафедральном вычислительном кла-

стере с использованием технологии MPI (*Message Passing Interface*, программный интерфейс передачи сообщений).

Разработанное СПО и методики (приёмы выявления скрытого параллелизма и его параметров в произвольных алгоритмах, способы построения рациональных планов выполнения параллельных программ на заданном поле вычислителей) ряд лет применяются при обучении студентов в указанных университетах России и позволили повысить компетенции учащихся в области параллелизации обработки данных.

Литература

[1] AlgoWiki. Открытая энциклопедия свойств алгоритмов. Под ред.: Воеводин В., Донгарра Дж. URL: `http://algowiki-project.org` (дата обращения: 15.04.2022).

[2] Воеводин В. В., Воеводин Вл. В. Параллельные вычисления. — СПб.: БХВ-Петербург, 2004. — 608 с.

[3] Баканов В. М. Управление динамикой вычислений в процессорах потоковой архитектуры для различных типов алгоритмов. // Журнал «Программная инженерия», — М.: 2015, № 9, с. 20–24.

[4] Федотов И. Е. Параллельное программирование. Модели и приёмы. — М.: СОЛОН-Пресс, 2018. — 390 с.

[5] V. M. Bakanov. Software complex for modeling and optimization of program implementation on parallel calculation systems. Open Computer Science, 2018, 8, Issue 1, Pages 228–234, ISSN (Online) 2299-1093, DOI: `https://www.degruyter.com/document/doi/10.1515/comp-2018-0019/html`

[6] Гэри М., Джонсон Д. Вычислительные машины и труднорешаемые задачи. — М.: Мир, Книга по Требованию, 2012. — 420 с.

[7] Иерузалимски Роберту. Программирование на языке Lua. — М.: ДМК Пресс, 2014. — 382 с.

Яковлев Виктор Вадимович
Москва, Московский Физико-Технический Институт (НИУ)
Проект: yajudge https://github.com/victor-yacovlev/yajudge

Ещё одна тестирующая система

Аннотация

Тестирующая система ejudge, разработанная около 20 лет назад — стандарт для автоматизации проверки задач по программированию как при проведении олимпиад, так и в учебном процессе. За годы эксплуатации системы ejudge в учебном процессе на курсах, связанных с операционных системами, был накоплен опыт использования, который привёл к созданию новой тестирующей системы. Новая система решает такие проблемы ejudge, как сложность администрирования и поддержки системы; отсутствие интерфейса преподавателя для мобильный устройств. Кроме того, улучшена изоляция тестирования с использованием Linux namespaces и cgroup, что позволяет выполнять тестирование задач на реализацию сетевых сервисов и межпроцессного взаимодействия. В настоящее время система успешно проходит апробацию на курсе по операционным системам в магистратуре МФТИ.

Предпосылки разработки

Для организации приёма заданий на курсе «Архитектура компьютеров и операционные системы» (АКОС) ФПМИ МФТИ [1] четвёртый год подряд используется тестирующая система Ejudge [2]. Несмотря на то что эта система изначально была создана для проведения олимпиадных мероприятий по программированию, она является достаточно гибкой в плане конфигурирования, поэтому возможно её использование и в учебном процессе.

Студентам каждую неделю открываются от 2 до 4 задач по самым разным темам начиная с основ низкоуровневого программирования, в том числе с использованием ассемблеров AArch64 и x86-64, до сетевого взаимодействия и использования сторонних библиотек. Для каждой учебной группы создаётся отдельный контест, в который регистрируются студенты, семинарист и учебный ассистент группы. Каждый контест имеет свои настройки дедлайнов, которые зависят от дня недели, в который проводятся занятия определённой группы.

Настройки задач и тесты — общие, это организовано через механизм символических ссылок.

Проверка большинства задач может быть организована штатными средствами ejudge, но есть и исключения, которые требуют нестандартной реализации, например: использование кросс-компиляции и запуска qemu, тестирование низкоуровневых сетевых задач, требующих канальный уровень сети, либо нестандартное взаимодействие с тестируемыми программами.

За годы использования ejudge приходится постоянно следить за работоспособностью тестирующей системы, поскольку нестандартные сценарии менее надёжны, чем хотелось бы. Кроме того, периодически приходится выполнять конфигурирование новых контестов, например, в начале каждого нового модуля, либо при добавлении новых задач. Учитывая тенденцию многих вузов по ежегодному увеличению количества студентов на IT-специальностях, и, соответственно, количества учебных групп, поддержка системы становится всё более трудоёмкой.

Для постепенной замены ejudge разрабатывается новая тестирующая система yajudge (Yet Another Judge), которая предназначена для использования в первую очередь именно в учебном процессе, и должна иметь следующие преимущества по сравнению с ejudge:

1. Упрощённое конфигурирование задач. Сейчас в ejudge используются конфигурации из файлов INI-формата, тексты заданий в формате XML, а если необходимо нестандартное тестирование или взаимодействие, то нужно писать вспомогательные программы на языке Си. При этом ошибки, например, в XML-структуре файлов условий, приводят к их неработоспособности с отсутствием подробной диагностики ошибки, а отлаживать нестандартное тестирование или взаимодействие достаточно трудоёмко.

2. Упрощённое конфигурирование курсов. В ejudge для создания нового контеста приходится избыточно дублировать одни и те же настройки, например, добавление нового преподавателя подразумевает не только добавление пользователя в таблице БД, но и правку конфигурационного XML-файла. Добавление же новой задачи в контест, может приводить к страшным последствиям, если задача добавляется не самой последней, поскольку ejudge

автоматически сопоставляет ID задач с тем, в каком порядке они встретились в конфигурационном файле.

3. Возможность аналитики отправленных решений, наиболее важная практическая задача в этом — анализ решений на плагиат. В системе ejudge отправленные решения хранятся в виде файлов, пути к которым выясняются из БД, что усложняет реализацию антиплагиата.
4. Большая степень изоляции, по сравнению с ejudge, при тестировании решений задач. Ввиду специфики изучаемого в курсе предмета, студенты могут использовать механизмы межпроцессного взаимодействия и взаимодействовать с сетевыми интерфейсами. Изоляция только на уровне отдельного пользователя и установка ресурсных лимитов не достаточны для безопасного выполнения программ, поскольку это не спасает, например, от посылки сигнала процессу с PID=-1, или открытию большого количества сетевых портов, которые затем не закрываются.
5. Возможность тестирования задач на разных системах, поскольку в курсе, помимо архитектуры x86, рассматривается архитектура ARM (AArch64 в последней итерации). Хотя тестирование с использованием эмулятора qemu и кросс-компилятора позволяет решать эту задачу, этот подход нельзя назвать идеальным.
6. Более современный пользовательский интерфейс. В частности, некоторые виды взаимодействий с системой (просмотр посылок, ревью кода) не требуют использования компьютера и могут выполнятся с мобильных устройств (планшеты и смартфоны), но интерфейс ejudge для этого не приспособлен.

Система yajudge

Система yajudge [3] состоит из центрального *мастер*-сервера, который должен быть установлен на машине с выделенным IP-адресом. Этот сервер предназначен для взаимодействия с *клиентами* посредством веб-интерфейса (также возможно применение «толстого» клиента), хранит сведения о пользователях, курсах и посылках в базе данных PostgreSQL, и раздаёт задания *грейдерам*, — отдельным сервисам, которые выполняют тестирование посылок задач.

Грейдер может быть запущен как в единственном экземпляре на том же физическом сервере, что и мастер-сервер, так и на отдель-

ных выделенных серверах, в том числе с динамическими IP-адресами, различными операционными системами (хотя только в Linux реализована изоляция запуска), и разной архитектурой процессоров. При подключении к мастер-серверу грейдер сообщает ему свою ОС и архитектуру процессора, поэтому мастер-сервер может раздавать грейдерам разные задания, в зависимости от требований отдельных задач.

Учебные курсы и задачи

Содержание заданий и учебных курсов хранятся в виде обычных файлов, которые находятся на машине с установленным мастер-сервером. Файлы, относящиеся к задачам, собраны в единый пул задач, у каждой задачи есть свой текстовый идентификатор, название которого совпадает с именем подкаталога задачи.

Содержание курсов хранится также в виде файлов, но отдельно от пула задач. Это сделано по двум причинам. Во-первых, контент курсов может иметь отличную от контента задач доступность, например, курс — общедоступный, а задачи — закрытые. Во-вторых, это позволяет достаточно гибко составлять разные курсы исходя из их продолжительности и целевой аудитории, используя при этом подмножества задач из единого пула. Курсы структурированы: разделы и отдельные уроки курса, которым соответствуют отдельные подкаталоги. Помимо ссылок на задачи из общего пула, курсы могут иметь материалы для изучения.

Для описания структуры курсов и параметров задач, как и для конфигурации системы в целом используется формат YAML [4], который является более компактным чем XML или JSON. Формат отдельных тестов для задач полностью позаимствован у системы ejudge, а для не стандартных тестирующих программы и генераторов тестов используется язык Python. Более подробное описание интерфейса приведено в демо-курсе [5].

Всё, что связано с динамически изменяемыми данными: пользователи, запуски курсов и посылки, включая решения заданий, хранятся уже в базе данных PostgreSQL.

Тестирование решений

Тестирование решений реализуется отдельным сервисом, который выполняет стадии проверки стиля кода, компиляции, запуска реше-

ний и сравнения с эталоном либо с использованием нестандартного проверяющего скрипта. Поддерживается тестирование программ на языках программирования Си/C++, языке ассемблера и скриптовых языков, например, Bash или Python.

Перед началом тестирования создаётся изолированная файловая система с использованием OverlayFS [6]. Нижние слои этой файловой системы состоят из:

1. Корневой файловой системы Linux-дистрибутива, который заранее подготовлен администратором. Это может быть, в простейшем случае, либо Alpine Linux [7], либо система, установленная в отдельный каталог с помощью `debootstrap` или других инструментов.
2. Каталога задачи, содержащего тесты и другие дополнительные файлы. Если используется генерация тестов, то она проводится на предварительном этапе, до начала тестирования. Генерация тестов полезна, например, для генерации случайных данных, либо для создания больших файлов.

Верхний слой файловой системы предназначен для создания исполняемого файла из исходного решения и сохранения результатов работы программы.

Полученная файловая система становится корневой для выполнения компиляции и прогона тестов. Таким образом, гарантируется, что отправленное студентом решение не будет иметь доступ к тем файлам, которые находятся вне задачи и специально выделенной файловой системы дистрибутива Linux.

Помимо изменения корня файловой системы, с помощью механизма Linux Namespaces [8] производится изоляция процесса. Наиболее значимые подсистемы, которые изолируются:

1. Точки монтирования файловых систем. Это, во-первых, необходимо для изменения корневой файловой системы без прав администратора, а во-вторых, делает безопасным тестирование решения задач на реализацию файловых систем с помощью FUSE.
2. Сетевые интерфейсы. Необходимо для задач на сетевое взаимодействие, которое становится возможным только с той вспомогательной программой, которая входит в поставку задачи.
3. Пространство процессов (PID). Это исключает возможность взаимодействия с процессами, не относящимися к задаче, используя механизм сигналов.

В качестве дополнительной меры безопасности можно запретить использование определённых функций стандартной библиотеки языка Си. Это реализуется на стадии компиляции решения использованием механизма подмены функций (опция `wrap` линковщика). Такое решение не является идеальным, и, возможно, будет заменено на использование механизма eBPF [9].

Для контроля лимитов запускаемого процесса используются два механизма, дополняющих возможности друг друга: ulimit, и механизм Cgroup v2 [10], который реализован в современных ядрах Linux.

Механизм Cgroup v2 позволяет более точно контролировать некоторые ограничения выполнения, чем классический механизм лимитов в UNIX. В частности, он даёт гарантии на количество одновременно запущенных процессов в рамках одного контейнера, что является важным для курса по операционным системам.

Пользовательский интерфейс системы

Пользовательский интерфейс системы реализован с помощью кросс-платформенного фреймворка Flutter [11], который предназначен для мобильной разработки, но также позволяет собирать программы под основные десктопные операционные системы (Linux, Windows, macOS), и в одностраничные Web-приложения.

Взаимодействие пользовательского интерфейса с мастер-сервером осуществляется, как и все межсервисные взаимодействия в yajudge, через протокол gRPC [12], который является более производительным, чем взаимодействие через REST+JSON. Для Web-версии клиентского приложения используется протокол gRPC-Web, — специальный вариант протокола gRPC для работы в условиях ограничений браузеров, и на стороне сервера необходим специальный прокси-сервер: либо Envoy [13], либо grpcwebproxy [14], входящий в поставку yajudge.

Итоги

Основной функционал системы реализован, и в настоящее время она проходит успешную апробацию на семестровом курсе по Операционным системам в магистратуре ФПМИ МФТИ. Это первая итерация практического использования системы, во время которой были устранены обнаруженные дефекты.

Пока ещё не реализована функциональность для проведения Code Review, которая необходима для использования системы в бакалавриате. Предполагается реализовать её к началу осеннего семестра 2022 года, после чего провести очередную итерацию практического тестирования.

Литература

[1] Материалы курса АКОС ФПМИ МФТИ https://github.com/victor-yacovlev/mipt-diht-caos

[2] Система Ejudge https://ejudge.ru

[3] Система Yet Another Judge https://github.com/victor-yacovlev/yajudge

[4] Формат YAML https://yaml.org

[5] Демо-курс yajudge https://demo.yajudge.ru/

[6] OverlayFS https://www.kernel.org/doc/html/latest/filesystems/overlayfs.html

[7] Alpine Linux https://alpinelinux.org

[8] Linux Namespaces https://man7.org/linux/man-pages/man7/namespaces.7.html

[9] eBPF https://ebpf.io

[10] Linux Control Groups v2 https://www.kernel.org/doc/Documentation/cgroup-v2.txt

[11] Flutter https://flutter.dev

[12] gRPC A high performance, open source unversal RPC framework https://grpc.io

[13] Envoy Proxy https://www.envoyproxy.io

[14] grpcwebproxy https://github.com/improbable-eng/grpc-web

Андреев Александр Николаевич, Яковлев Виктор Вадимович
Москва, Московский Физико-Технический Институт (НИУ)

Проект: HellOS https://github.com/carzil/hellos

Курс по разработке операционной системы на базе HellOS

Аннотация

В прошлом году была анонсирована операционная система HellOS, предназначенная для целей обучения разработки операционных систем. Перед студентами ставятся задачи по развитию этой небольшой операционной системы в рамках мини-проектов, и за последний год была реализована новая функциональность: поддержка 64-битных процессоров x86, 4-уровневая адресация памяти, Multiboot 2, инструкции syscall/sysret вместо прерываний. Также запланированы улучшения поддержки SMP а архитектуры AArch64.

Использование в учебном процессе

В этом году состоялась новая итерация курса [2], которая потребовала определённых правок в ядре HellOS. Основные изменения, которые реализованы:

1. Поддержка 64-битных процессоров x86. 32-битная версия x86 имеет ряд механизмов, которые не используются современными ОС (например, hardware multitasking и non-flat segmentation). Изучение этих рудиментарных особенностей усложняет понимание материала лекций, семинаров и домашних заданий, не добавляя полезных знаний. К тому же на сегодняшний день 32-битные процессоры на основе x86 — довольное редкое явление.
2. 4-уровневая адресация памяти. С появлением 64-битной адресации в HellOS требуется обеспечить также 64-битное адресное пространство. Поэтому в ОС, а также в материалы курса была добавлена поддержка 4-уровневой адресации памяти. Кроме этого, вся подсистема управления памятью была структурирована и переработана.
3. Инструкции syscall/sysret. Исторически самый первый способ вызова сисколлов — через программные прерывания (int 0x80). Однако это довольно медленный механизм, требующий много

проверок со стороны процессора. Вместе с реализацией поддержки x86-64 стало возможным использование современных способов для вызова сисколлов — через инструкцию syscall.

Реализация системы и учебного курса

Система HellOS реализована на языке C для архитектуры x86-64 и предназначена для работы в виртуальной машине QEMU, эмуляторе BOCHS и на обычных компьютерах. Представляет собой простое монолитное ядро, расположенное в верхних адресах памяти.

В HellOS реализованы базовые функции для работы с примитивами x86: обработка прерываний, работа с виртуальной памятью и другая базовая функциональность операционной системы. Реализованы функции парсинга и валидации ACPI таблиц. Для управления обработкой прерываний APIC/LAPIC HellOS имеет базовый не fair-share планировщик процессов, который не поддерживает приоритеты процессов. Все эти расширения остаются в качестве лабораторных работ.

Также в HellOS есть драйвер для ATA-совместимых устройств, в качестве механизма трансфера данных используется DMA вместо устаревшего PIO. Поверх драйвера реализована виртуальная простая виртуальная файловая система, на текущий момент поддерживается только ext2.

Основываясь на архитектуре HellOS, курс разбит на следующие темы:

1. Процесс загрузки и bootloader'ы.
2. Прерывания процессора.
3. Виртуальная память.
4. User-space процессы, системные вызовы и переключение контекста.
5. Многопроцессорность.
6. Работа с HDD (режимы PIO и DMA).
7. Базовые принципы работы файловых систем на примере ext2.

Итоги

Реализована операционная система, которая опубликована на GitHub под лицензией MIT. Система уже второй год подряд успешно

используется в курсе «Архитектура операционных систем» в качестве наглядного пособия, а также в качестве платформы для домашних заданий.

Литература

[1] СПО: от обучения до разработки: Объединённая конференция: сборник тезисов конференции, Переславль-Залесский, 15–18 июня 2021 года. — Москва: ООО «МАКС Пресс», 2021. — 216 с. — ISBN 978-5-317-06632-1

[2] Материалы курса «Архитектура операционных систем» ФПМИ МФТИ `https://github.com/carzil/mipt-llp-2022`

ПроектLJV/JOL, Jedis-Mock, JSyntrax, Xylophone

Олег Ивченко, Иван Пономарёв
Долгопрудный, Московский физико-технический институт (национальный исследовательский университет)
https://github.com/atp-mipt

Опыт контрибьютинга в свободное ПО как практической работы студентов в процессе обучения разработке

Аннотация

Традиционными заданиями для самостоятельной работы при обучении программированию является решение стандартных задач или разработка студенческих проектов. Хотя данный подход обеспечивает тренировку студентов на достаточно лёгких задачах и объективную оценку результатов работы, сам процесс и результаты самостоятельной работы очень сильно отличаются от того, с чем встречается разработчик на практике. Мы на нашей кафедре уже несколько лет экспериментируем со студенческими проектами, результатом которых является коммит в свободное ПО, принятый мантейнерами. В докладе мы расскажем, какие преимущества даёт подобный подход по сравнению с решением стандартных задач, с какими трудностями приходится сталкиваться, чем надо руководствоваться при выборе подходящего проекта свободного ПО, и кратко расскажем о нескольких примерах проектов: LJV/JOL, Jedis-Mock, JSyntrax, Xylophone

Самостоятельные задания, обычно выдаваемые студентам, обучающимся технологиям разработки, сильно отличаются от того, с чем им придётся столкнуться на практике. Вот перечень аспектов, которые обычно не отрабатываются при обучении студентов:

1. Совместная работа многих разработчиков с использованием GitHub и подобных систем.
2. Процесс разработки: issues, pull requests, code review.
3. Технологический аспект: CI-системы, quality gates, ситуации, когда изменения не принимаются статическим анализатором.
4. Работа с существующим кодом, не идеальность существующего кода, необходимость рефакторинга перед внесением изменений.
5. Необходимость писать автотесты (культуре и навыкам написания автотестов практически не уделяется внимание при обучении разработчиков).

Закрыть эти пробелы можно с помощью практики разработки на реальных Open Source проектах. На протяжении трёх лет на кафедре АТП МФТИ мы проводим эксперимент, при котором студентам, обучающимся технологиям разработки на Java, имеющим базовые знания и желающим получить более сложное задание, предоставляется возможность в течение семестра работать над каким-либо из проектов с открытым исходным кодом в виде альтернативы к стандартному домашнему заданию.

Проекты для студенческой работы отбираются преподавателем, студентам предоставляется выбор из нескольких альтернатив.

Разумеется, не каждый проект подойдёт для студенческой работы, но огромное количество и разнообразие открытых проектов даёт возможность отобрать подходящие.

На наш взгляд, подходящий учебный проект должен удовлетворять следующим условиям:

1. Небольшой объём: один инженер (преподаватель) должен быть в состоянии «охватить» все его аспекты. Объём наших проектов не превышал 10 KLOC.
2. Наличие базы пользователей: это источник сообщений об ошибках и желаемых функциональных возможностях
3. Отсутствие процедур обсуждения изменений: длинные сроки обсуждений и согласований, как правило, делают невозможным произвести учебный цикл в течение семестра.

4. Преподаватель должен входить в число мантейнеров проекта и должен иметь право интегрировать изменения в основную ветку. (Это происходит после code review изменений от студентов и также зачастую после дополнительных изменений кода, сделанных самим преподавателем.) Хорошо, если преподаватель также имеет возможность принимать решение о релизе новых версий проекта: быстрое попадание изменений в релиз является фактором, мотивирующим студентов.

Помимо выбора проекта, важен выбор задач к реализации на проекте. Успешность реализации задачи является основанием для оценки работы студента, но должна существовать возможность объективно оценить успешность. По нашему опыту хорошо подходят следующие задачи:

1. **Исправление ошибок из числа открытых тикетов.** Обязательным требованием является написание автотестов, воспроизводящих ошибки.

2. **Рефакторинг при наличии тестов.** Обсуждается желаемая архитектура, после чего студент производит улучшение кода, добиваясь «зелёной» сборки на CI. Иногда для таких задач требуется написание дополнительных тестов преподавателем.

3. **Переимплементация** известной функциональности. Наиболее успешными оказались проекты, в которых стояла задача переимплементации, существующей на другом языке / фреймворке. В этом случае легко сверять результат с «эталоном» и создавать тесты.

Неудачными являются задачи, для которых нет абсолютно точно сформулированных требований. Студент не готов к тому, чтобы выполнять задачи, включающие в себя дизайн решений. Отсутствие точных требований означает также и отсутствие критериев успешности выполнения, что затрудняет оценивание студента.

«Опасными» являются задачи, для которых нет понятного способа решения, и, как следствие, требующие исследований со стороны студента. Исследования могут завершиться успехом, но в случае отрицательного результата часто невозможно определить причину: виной недостаточная работа студента или объективное отсутствие решения.

Как правило, в течение семестра студенты получают задачи разного типа: как простые, так и «исследовательские».

Перечень проектов

LJV: Lightweight Java Visualizer. `https://github.com/atp-mipt/ljv`. Объём 1.5 KLOC. Является оживлением заброшенного проекта LJV для визуализации внутренних структур данных. Проект набрал с нуля 200+ звёзд на гитхабе, публикации и выступления по LJV были довольно успешны.

Jedis-mock: тестовый мок Redis на Java. `https://github.com/fppt/jedis-mock`. Объём 8KLOC. Jedis-mock является частичной реимплементацией Redis на Java для нужд тестирования. Проект обладает небольшой пользовательской базой. Успешно проведён массированный рефакторинг, реализована новая функциональность, закрываются запросы пользователей.

JSyntrax: визуализация формальных языков в виде railroad-диаграмм в форматах PNG и SVG `https://github.com/atp-mipt/jsyntrax/`. Объём 3KLOC. Является переимплементацией старого заброшенного проекта Syntrax с Python на Java. Проект поддержан в AsciiDoctor.

Xylophone: `https://github.com/CourseOrchestra/xylophone`. Объём 3KLOC. Решает задачу создания отформатированных Excel-документов на основе данных, представленных в XML. Библиотека нужна для многих организаций, генерирующих печатные формы, отчёты и тому подобное. Проект старый, имеет пользовательскую базу.

Выводы

Практика студенческой работы над Open Source проектами в целом показывает очень позитивные результаты. Для того чтобы работа была успешной (студенты ощущали, что делают полезную работу, результаты работы попадали в релизы), проекты и задачи к реализации на проектах должны быть выбраны по ряду признаков, перечисленных в данном докладе.

Литература

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

Алексей Федорчук
Москва, Независимый автор
Проект: Alv's story https://www.alvstory.ru/

Linux для пращуров и внуков

Аннотация

Доклад посвящён объединённому проекту «Linux для сочинителей» и «Linux для пенсионера». Они основывались на базовой Ubuntu с рабочим окружением Cinnamon и Gtk-приложениями. Однако и с базисом, и с рабочей средой предвидятся проблемы. Поэтому решено было заблаговременно подбирать альтернативы. Среди кандидатов были: MX Linux и Netrunner Core (apt), Alt KDE StarterKit и PCLinuxOS KDE Darkstar (apt-rpm), EndeavourOS (pacman) и NuTyX (cards). Все они отвечают требованиям, но с различными (в каждом случае) оговорками. Меньше всего оговорок появилось в отношении MX Linux. В дубль-составе NuTyX, как система не столько для пенсионеров, сколько для их внуков.

Автор доклада на протяжении ряда лет занимается двумя темами, связанными с СПО, которые условно можно назвать «Linux для сочинителя» и «Linux для пенсионера». Оба основывались на базовой Ubuntu (т.н. minimal), рабочей среде Cinnamon и ограниченном наборе Gtk-приложений для работы с текстами и их иллюстрирования. Однако со всеми тремя компонентами наметились проблемы.

Прекратили развитие PPA-репозитории, которые служили источником пакетов среды Cinnamon актуальных версий. Каковые в каждый момент времени гарантированно имелись только в репозитории Linux Mint CE, который (в версии 20.3 una) и было решено выбрать в качестве паллиативной основы для наших обеих тематических систем.

Поэтому решено было заблаговременно подбирать альтернативы, которые а) не основывались бы на Ubuntu, и б) рабочая среда и приложения были бы не зависимы от Gtk. В ходе отбора кандидатов выяснилось, что требования к системам сочинительской и пенсионерской практически идентичны. Что и неудивительно: мемуары и воспоминания — один из любимых занятий пенсионеров, по крайней мере, нашей профессии. А как показало общение в сети — и не только нашей. А что это иное, как не частный случай сочинительского ремесла? Поэтому

две озвученные выше темы были слиты в единый проект: «Linux для пращуров и внуков».

Затем начался подбор альтернатив. Рабочая среда определилась как KDE — единственная, сравнимая с Cinnamon по возможностям и их настройкам. А от дистрибутивов, помимо поддержки KDE, требовались: быстрота установки, лёгкость освоения, простота применения и сопровождения, в частности, внятные средства управления пакетами. «Свежесть» версий также учитывалась, хотя допускалась и «вторая». Кроме того, желательна компактность, то есть целенаправленный отбор приложений, дабы с самого начала не устрашить нашего потенциального пенсионера их избытком.

Требования вытекают из того, что у пенсионеров мало времени. Не из-за занятости: его у них просто мало, и надо успеть сделать задуманное. Отсюда и субъективный фактор — достаточное знакомство с кандидатами (или родственными дистрибутивами) автора этих строк: ему не хотелось тратить время на знакомство с системой абсолютно новой.

С этих позиций было рассмотрено много систем из числа тех, что автор видел, работал, а иногда, про которые, даже писал. Некоторые системы показались слишком сложными для начинающего пенсионера. А вариации на темы Red Hat или Suse в свете текущих событий были отвергнуты.

В итоге в «сухом остатке» оказались: MX Linux и Netrunner Core (дериваты Debian stable. apt), Alt StarterKit и PCLinuxOS Darkstar (некогда произошедшие от Mandrake, apt-rpm), EndeavourOS (дериват Arch'a, pacman) и «сирота»NuTyX (cards). Все они, хоть и с оговорками, отвечают требованиям. Но, наряду с бесспорными достоинствами, обладают особенностями, которые, не будучи недостатками, могут оказаться таковыми в рамках нашей задачи.

Рекордсмен по компактности — Netrunner Core (iso-образ 1,3 ГБ — и это с KDE!). И он же, ввиду своей основы, оказался аутсайдером по «свежести», которая у него на грани «третьей». Компактные версии Alt и PCLinuxOS (1,3 и 1,7 ГБ, соответственно) развиваются по модели semi-rolling, версии пакетов обновляют интенсивно. Но apt-rpm выглядит по сравнению с современным apt несколько архаично. Чистый rolling, EndeavourOS, при относительной компактности (1,9 ГБ), — чемпион по актуальности базиса, среды и приложений. Оборотная сторона чего — некоторая сложность для начинающего пенсионера.

Меньше всего оговорок было в адрес MX Linux. Он не может похвастаться ни «наисвежайшим софтом», ни особой компактностью (2,3 ГБ). Однако имеет собственный репозиторий, в том числе AHS (Advanced Hardware Support), обеспечивающий поддержку новейшей аппаратуры, и — комплекс универсальных фирменных утилит. Ввиду чего MX занял 1-е место в нашем гандикапе.

В дубль-составе — NuTyX (в варианте base — 700 МБ). Он сложноват для начинающего пенсионера, но на его основе легко создать очень простую в освоении и использовании систему. Так что это Linux не для пенсионеров, а для их внуков, готовых затратить некоторые усилия и поделиться их результатами с близкими. Принадлежи автор к поколению не дедов, а хотя бы отцов — он бы занялся этим сам...

Анатолий DOC Якушин
Москва

Протестантская этика Макса Вебера и дух свободного программного обеспечения

Аннотация

Экономический и социальный мониторинг свободных проектов последних лет убедительно показывает, что разработка и дистрибуция СПО сегодня — это ещё одна форма коммерческого программного обеспечения, в оборот которого вовлечены все ведущие производители ПО. Исследование анализирует те изменения, которые произошли в сообществе свободных разработчиков за последнее десятилетие.

Появление феномена свободного программного обеспечения (далее СПО) в создании и развитии информационных технологий положило начало новым моделям и теориям социальной организации и нематериального труда. Побудительные мотивы родоначальников движения прямо проистекают из положений хакерской этики, зародившейся в недрах MIT и других высших учебных заведений США в 70-е годы прошлого века [1]. Традиционные социологические исследования обычно прямо противопоставляют «этику хакеров» и «протестантскую трудовую этику», впервые описанную в трудах М. Вебера ([2], [3]). Однако углублённый анализ мотивации участников раннего

СПО показывает, что между двумя этическими полюсами нет существенного противоречия, лишённая религиозной составляющей «протестантская этика» вполне соответствует побудительным мотивам основоположников движения. Раскрытие внутренних мотивов является необходимым для отслеживания изменений, происходящих в движении СПО на протяжении всего его существования и построении прогнозов дальнейшего развития сообщества.

К сожалению, эпоха тотальной коммодификации [4] не могла не отразиться на составе участников движения СПО, их поведении и основных этических представлениях. Если ранние хакеры являли собой андерграунд IT-отрасли и их мало интересовали нравственные представления большинства, то сегодня участники движения оказываются под неоспоримым влиянием политических проблем, торговых войн, таких движений, как #MeToo, BLM и аналогичных. Порой это приводит к появлению весьма показательных монстров, например, The Hippocratic License [5] и иных проявлений нравственных взглядов современных разработчиков. Достаточно вспомнить внушительный список весьма уважаемых членов сообщества, вынужденных его покинуть по весьма надуманным «этическим» поводам.

Может создаться впечатление, что анализ этических представлений разработчиков СПО и побудительных мотивов их участия в деятельности сообщества имеет мало практической пользы. Однако следует понимать, что СПО по модели Р. М. Столлмана всегда было очень специфическим набором этических норм, где доступ к исходному коду был ключом к свободе и защищал эту свободу исключительно копилефт. Однако сегодня мы наблюдаем устойчивое снижение доли копилефтных лицензий, если в 2012 году их было 59% среди всех открытых проектов, то в 2019 году только 33% [6].

Появление на рынке игроков СПО таких гигантов, как Microsoft, Google и других, активное привлечение к работе в свободных проектах full time разработчиков на средства этих гигантов неизбежно приведёт к атаке на копилефтные лицензии и приведение их к виду, устраивающему современную IT-индустрию. Сможет ли выстоять копилефт в это непростое время, покажет будущее. Именно решение этого вопроса определит ландшафт свободного ПО в ближайшее десятилетие.

Литература

[1] Стивен Леви. Хакеры: Герои компьютерной революции. Penguin USA. 2001. ISBN: 0141000511

[2] Pekka Himanen. The Hacker Ethic and the Spirit of the Information Age. Publisher: Random House Inc. 201 East 50th Street New York, NY United States ISBN:978-0-375-50566-9

[3] Max Weber. Die protestantische Ethik und der 'Geist' des Kapitalismus. 1905.

[4] Фелпс, Э. Массовое процветание: Как низовые инновации стали источником рабочих мест, новых возможностей и изменений / пер. с англ. Д. Кралечкина; науч. ред. Перевода А. Смирнов. — М.: Изд-во Института Гайдара; Фонд «Либеральная Миссия», 2015. — 472 с.

[5] Mark Radcliffe. Top 10 FOSS legal developments in 2019. https://www.synopsys.com/blogs/software-security/top-10-open-source-legal-issues-2019/

[6] Cliff Saran, Adrian Bridgwater. The future of open source licences is changing. https://www.computerweekly.com/feature/The-future-of-open-source-licences-is-changing

Панюкова Александра Анатольевна, Арлинский Павел Дмитриевич, Петров Денис
Москва, ГАПОУ КП№11, ООО «ШЭРИКС»

Проект: ShariX https://sharix-app.org/

Студенческий проект: программные продукты на основе платформы ShariX

Аннотация

В статье рассказывается о платформе ShariX и проектах на её основе. Это студенческая разработка, создаваемая и развивающейся благодаря инициативе преподавателя и студентов, решивших продолжить своё участие в проекте в результате прохождения обучения по серии курсов («Информационная этика», «Инновационные технологии» — раздел «IT решения», «Основы предпринимательства»).

Платформа ShariX является студенческой разработкой, создаваемой и развивающейся благодаря инициативе преподавателя и студентов, решивших продолжить своё участие в проекте в результате прохождения обучения по серии курсов («Информационная этика», «Инновационные технологии» — раздел «IT решения», «Основы предпринимательства»).

Рассказ о проекте направлен в том числе на привлечение молодых специалистов в проект.

ShariX — это стартап в области набирающих популярность шеринговых сервисов. ShariX проектируется как платформа, которая включает в себя базу пользователей и инструменты для построения sharing-сервисов. На её основе можно создавать собственные сервисы, например, на основе базовых сервисов с открытым исходным кодом.

ООО «ШЭРИКС» — специально организованная для данного проекта компания. Она разрабатывает программный продукт, на основе которого организуется база данных пользователей, позволяющая выполнять некоторый набор операций с ними для успешного оказания услуг клиентам. Созданы шаблоны для построения дочерних сервисов различных услуг на основе общей концепции. Предполагается, что ShariX является единой точкой входа для потребителя как плательщика, и определяет схему распределения платежа в конечной цепочке исполнителей. Принимая оплату от конечного исполнителя и распределяя средства по заранее согласованной схеме услуги, она является гарантом для всей цепочки в получении оплаты (всем достаточно убедиться в добросовестности одного контрагента, а не произвольного количества).

Цепочка исполнителей внутри услуги может быть любой длины, но у каждого контрагента есть своё назначение в ней. Чтобы проще было понять суть цепочки, приведём пример на основе сервисов заказа такси.

Водитель — конечный исполнитель, он возит клиента (выполняет работу).

Партнёр сервиса — юридическое лицо или индивидуальный предприниматель, который оказывает услугу перевозки клиенту и фактически несёт ответственность за безопасность конкретной поездки и т.п.

Сервис («дочерний сервис» в описании выше) — сервис перевозки пассажиров, который определяет стандарты качества, соответствие им, конкретную форму и особенности услуги, её продвижение и пози-

ционирование для пользователей, особенности формирования тарифов и так далее. Но для того чтобы технологически соединить водителя и пассажира, привлекаемых в наиболее подходящей для всех сторон форме, он обращается к программному обеспечению платформы.

Концепция предполагает возможность создавать сервисы, которые предлагают составные услуги, добавляют свою услугу к цепочке услуг других сервисов, например, присылают водителя, который арендует какой-то конкретный автомобиль.

Для того чтобы популяризировать создание сервисов на основе платформы, предполагается сделать доступным исходный код шаблонного сервиса (проект ShariX Open). Он спроектирован таким образом, что при определённых настройках может функционировать и без привязки к основной платформе при самостоятельной реализации системы оплаты (при её необходимости), то есть являются независимым программным продуктом.

Особое внимание стоит уделить особенностям выбора лицензии для публикации данного продукта. С одной стороны, разработчикам с точки зрения развития продукта интересен наиболее сильный копилефт.

С другой стороны, компаниям, которые будут строить свои уникальные решения на основе данного продукта, интересно было бы получать конкурентное преимущество не только посредством средств индивидуализации, рекламных кампаний и особенностей оказания услуг, но и визуальным представлением интерфейса в целом, а также возможностью интеграции собственных дополнительных решений.

Ввиду экономической целесообразности и интереса для партнёров сервиса, осуществляющих предпринимательскую деятельность, необходимо разделить программный код на части и ввести различный правовой режим для frontend и backend части.

Одной из подходящих для развития проекта с юридической точки зрения является GNU AGPLv3. Так как эта лицензия с сильным копилефтом, что в достаточной степени защищает правообладателя и предполагает возмездный характер отношений сторон. Также в ней содержится расширительное толкование распространения кода в случае предоставления доступа через интернет, что подходит для backend в целом и способствует развитию проекта.

При предоставлении права пользования открытой частью кода, отвечающую за внешний вид интерфейса, особенности расположения

его элементов, необходимо предоставить дочернему сервису, работающему в сети, на основании лицензии GNU GPLv3 способ сохранить за собой возможность использовать свой индивидуализированный продукт с сохранением и защиты прав на результаты интеллектуальной деятельности, так как данная лицензия также является копилефтной, но GPL разрешает производить модифицированные версии и применять их безо всякой передачи другим.

Шмырёв Н. В.
Москва, АЦ Технологии

Проект: Vosk `http://alphacephei.com`

Перспективы развития открытых проектов в области машинного обучения

Аннотация

Библиотека для распознавания речи «Воск» поддерживает более 20 языков и диалектов. Работает без доступа к сети на серверах и мобильных устройствах, поддерживает платформы Raspberry Pi, Android, iOS. Как создатели библиотеки мы хотим описать текущее состояние и перспективы развития небольших проектов машинного обучения.

Современную историю проектов машинного обучения можно разделить на следующие этапы:

– академические проекты (2000–2010);

– академические проекты в сотрудничестве с индустрией (2010–2017);

– крупные индустриальные проекты (2018–2022).

В настоящее время самые значительные проекты в машинном обучении создают крупные организации с мощными вычислительными возможностями: Facebook, Google, Microsoft, Baidu. Академическая разработка переживает кризис из-за ограниченного доступа к вычислениям, для исследований нужны мощные суперкомпьютеры с десятками GPU карт.

С другой стороны, из-за доступности данных и технологий появляются десятки открытых проектов от небольших компаний и команд, решающих сходную задачу. В области распознавания речи компании создают десятки систем распознавания речи, диалоговых систем,

систем компьютерного зрения, этим занимаются крупные компании: Google, Facebook, Nvidia, Microsoft, Mozilla, Baidu, Bytedance, Tencent, Clova, Сбербанк и так далее.

В таких условиях мы видим перспективу развития небольших проектов в следующих областях:

1. Интеграция и упаковка существующих решений. Из-за разнообразия огромное значение приобретает предоставление единого интерфейса для использования существующих библиотек. Важна и интеграция с другими программными компонентами, например, телефонией, веб-конференциями, средами разработки игр.

2. Специализация в предметных областях и использование дополнительной информации для более эффективного решения предметных задач. Большинство современных подходов направлены на создание наиболее общих решений, требующих серьёзных вычислительных затрат, что позволяет получать результаты в более специализированных подходах. Здесь же стоит рассмотреть специализированные аппаратно-программные решения.

3. Поддержка распространённых мировых языков. Большинство российских и китайских проектов ориентированы на внутреннее применение, что существенно снижает их потенциал. Западные проекты концентрируются на малых языках и специальных применениях, также ограничивая свою применимость. Проекты, поддерживающие основные языки, получают уникальное преимущество.

Дмитрий Маракасов
Москва

Проект: Repology https://repology.org/

Repology: мониторинг пакетных репозиториев

Аннотация

В докладе будет освещена история и основные возможности проекта Repology, ведущего мониторинг более 350 пакетных репозиториев с целью обнаружения новых релизов, координации работы мантейнеров пакетов и авторов ПО между собой, выявления уязвимых пакетов и пакетов, требующих обновления. Будут затронуты встретившиеся на пути проекта сложности и их решения, вылившиеся в побочные проекты, как-то универсальная библиотека сравнения версий libversion и агрегатор метаданных о Python модулях PyPICache.

В мире насчитывается несколько сотен независимых открытых пакетных систем, но исторически разработка большинства из них происходит в своеобразной изоляции, поскольку разница в форматах пакетов и рецептов сборки, организационных и идеологических решениях не способствует не только прямому обмену кодом, но даже сравнительному анализу пакетных баз. Тем не менее, значительная часть работы (к примеру, подготовка патчей и банальный мониторинг новых релизов и уязвимостей) многократно дублируется и кажется, что вся экосистема значительно выиграла бы, будь у мантейнеров быстрый доступ к конфигурациям аналогичных пакетов в других системах и возможность сравнения с ними, равно как и «online» информация по полноте и актуальности собственной пакетной базы.

В попытке проверить эту гипотезу появился проект Repology. Сервис работает уже более 5 лет и на данный момент собирает информацию из более чем 350 источников, включающих основные и дополнительные пакетные репозитории большинства Linux и *BSD дистрибутивов, сторонние пакетные менеджеры других систем (MacOSX, Windows, Android, AIX, Solaris), коллекции модулей языков программирования (PyPI, RubyGems, crates.io, CPAN и др.), а также новостные сайты и каталоги СПО.

Собранная информация приводится к единому формату, нормализуется и агрегируется до уровня upstream проектов, нивелируя разницу в том, как именно они опакечены в отдельных репозиториях. На

уровне проекта становится возможным сравнение номеров версий с определением актуальных и, соответственно, версий, требующих обновления, а из этой информации — составление широкого набора выборок, отчётов, atom лент и графических информеров в срезах по отдельным проектам, мантейнерам и репозиториям. Собирается и другая информация, например, описания и ссылки на upstream, рецепты пакетов, логи сборки и патчи.

Проект решает задачи сразу нескольких заинтересованных сторон. Для мантейнеров пакетов Repology позволяет:

- Получать полную и актуальную информацию о новых релизах проектов, пакеты для которых они поддерживают (доступную, например, сразу как список требующих обновления пакетов для конкретного мантейнера, или как ленту сообщений о новых релизах), об обнаруженных уязвимостях и других проблемах.
- Выявлять недостающие пакеты для популярных (по наличию в других дистрибутивах) или новых проектов.
- Находить контакты коллег-мантейнеров из других дистрибутивов для консультаций, обмена опытом и информирования о проблемах.
- Искать готовые решения проблем, например, в виде патчей или комбинаций флагов сборки.

Для дистрибутивов/репозиториев в целом:

- Иметь статистику по полноте и актуальности пакетной базы.
- Своевременно получать информацию об обновлениях и уязвимостях по всем пакетам, возможно, экономя собственные ресурсы на создании локальных инструментов для этой цели.
- Использовать предоставляемые Repology API готовые графические информеры в собственной инфраструктуре.

Для авторов ПО:

- Иметь возможно полное представление обо всей downstream инфраструктуре: в каких дистрибутивах и системах опакечен проект, под какие архитектуры собирается, с какими настройками и патчами, с какими версиями зависимостей и т.д.
- Поддерживать двустороннюю связь с мантейнерами пакетов, как помогая исправлять недочёты в пакетах, так и улучшая собственную систему сборки и документацию для упрощения развёртывания и эксплуатации.

- Использовать готовые графические информеры на страницах проекта, информируя пользователей о доступности и актуальности готовых пакетов.

На текущий момент поток обратной связи и статистика использования позволяют делать выводы о популярности и востребованности сервиса, а для кого-то он даже стал незаменимым инструментом. Развитие проекта продолжается как в сторону добавления новых возможностей (например, в планах добавить прямую поддержку VCS хостингов типа GitHub как источника информации о новых релизах), так и в сторону улучшения удобства сайта.

Антон Бондарев
Санкт-Петербург, Embox
http://embox.github.io/

Интернет вещей на базе СПО

Аннотация

В докладе рассматривается построение устройств интернета вещей на базе открытой ОС РВ Embox и открытых реализаций протокола MQTT, таких, как Eclipse Paho и подключение этих устройств в инфраструктуру на основе открытого брокера Eclipse Mosquitto.

Интернет вещей (Internet of Things (IoT)) проникает во всё большее количество сфер нашей жизни. Если раньше понятие IoT ассоциировалось в основном с «умными» чайниками или лампочками, весь «ум» которых заключался в возможности управления ими при помощи смартфона, то со временем, появилась возможность использовать механизм взаимодействия между умными устройствами и умные алгоритмы управления на основе этих данных, причём при минимальном участии человека. Кроме того идеи «Интернета вещей» нашли своё место в промышленности (промышленный «Интернет вещей» (IIoT)) и во многих других областях деятельности человека.

Для взаимодействия в IoT могут использоваться совершенно разные протоколы. На сегодняшний день самым распространённым в обычном IoT является MQTT[1] (message queuing telemetry transport).

[1] `https://mqtt.org/`

Данный протокол позволяет организовать общение между устройствами с помощью сообщений по принципу издатель-подписчик. Сеть MQTT подразумевает наличие брокера (сервера) и различных терминальных устройств (клиентов). Любое терминальное устройство может быть как издателем или подписчиком, так и совмещать обе эти роли.

Протокол MQTT имеет уже несколько версий стандартов. Кроме того, в оригинальную версию MQTT реализованную поверх TCP, добавили версию стандарта для работы по UDP MQTT-SN (sensor network).

Существуют различные реализации брокера MQTT как открытые так и проприетарные, разработанные на разных языках и имеющие различный функционал. На сегодняшний день наиболее популярным является Mosquitto[2]. Кроме того, существуют ряд облачных решений, доступных онлайн и уже готовых к использованию.

Существует множество библиотек разработанных на различных языках для реализации клиентской части устройств. Часть из них является специализированными и оптимизированы для запуска на одной встроенной RTOS или на конкретной платформе, например, Arduino или mbed. Подобные версии имеют ограниченный по сравнению с универсальными версиями функционал и не полностью соответствуют стандартам.

Большинство реализаций не являются специфичными для какой-то платформы и ориентированы на применение в универсальных платформах. Наиболее распространённым является проект Eclipse paho[3], имеющий реализации для нескольких языков, а C и C++ версии адаптированы под ряд специфичных платформ.

Embox[4] свободная ОС для встроенных систем. Основная идея — запуск ПО Linux на любых платформах, в том числе микроконтроллерах. Это, в частности, позволяет использовать для терминальных устройств универсальные версии клиентских библиотек MQTT, в частности, полноценные версии проектов с открытым кодом paho.mqtt.c[5] и paho.mqtt.cpp[6].

[2] https://mosquitto.org
[3] https://www.eclipse.org/paho/
[4] https://github.com/embox/embox
[5] https://github.com/eclipse/paho.mqtt.c
[6] https://github.com/eclipse/paho.mqtt.cpp

В системе сборки Embox предусмотрен механизм для упрощения работы с внешними проектами. Он позволяет описывать в Makefile аналог традиционных команд (`./configure; make; make install`). Пример `Makefile` для paho.mqtt.c

```
PKG_NAME    := paho.mqtt.c
PKG_VER     := v1.3.8
PKG_ARCHIVE_NAME := $(PKG_NAME)-$(PKG_VER).tar.gz
PKG_SOURCES := https://github.com/eclipse/paho.mqtt.c/archive/$(PKG_VER).tar.gz
PKG_MD5     := 71b9a3070f543afcb818a8c231be6684
PKG_PATCHES := patch.txt
include $(EXTBLD_LIB)
$(CONFIGURE) :
    cd $(PKG_INSTALL_DIR) && ( \
       CC=$(EMBOX_GCC) \
       CFLAGS="$(EMBOX_IMPORTED_CFLAGS)" \
       CPPFLAGS="$(EMBOX_IMPORTED_CPPFLAGS)" \
       cmake -DPAHO_WITH_SSL=FALSE \
           -DPAHO_BUILD_STATIC=TRUE \
           -DPAHO_BUILD_SHARED=FALSE \
           -DPAHO_BUILD_DOCUMENTATION=FALSE \
           -DPAHO_BUILD_SAMPLES=FALSE \
           -DPAHO_ENABLE_TESTING=FALSE \
           -DPAHO_ENABLE_CPACK=FALSE \
           -DCMAKE_C_COMPILER_WORKS=1 \
           -DPAHO_HIGH_PERFORMANCE=ON \
           $(PKG_SOURCE_DIR)/ \
    )
    touch $@
$(BUILD) :
    cd $(PKG_INSTALL_DIR) && ( \
       $(MAKE) MAKEFLAGS='$(EMBOX_IMPORTED_MAKEFLAGS)'; \
    )
    touch $@
$(INSTALL) :
    cp $(PKG_INSTALL_DIR)/src/libpaho-mqtt3c.a $(PKG_INSTALL_DIR)/libpaho-mqtt3c.a
    cp $(PKG_INSTALL_DIR)/src/libpaho-mqtt3a.a $(PKG_INSTALL_DIR)/libpaho-mqtt3a.a
    cp $(PKG_SOURCE_DIR)/src/*.h $(PKG_INSTALL_DIR)/
    touch $@
```

Для его использования в `Mybuild` используется следующее описание:

```
package third_party.mqtt.paho_mqtt_c
@Build(stage=1,script="$(EXTERNAL_MAKE)")
@BuildArtifactPath(cppflags="-I$(abspath
                $(EXTERNAL_BUILD_DIR))/third_party/mqtt/paho_mqtt_c/libpaho_mqtt3c/install/")
static module libpaho_mqtt3c {
    @AddPrefix("^BUILD/extbld/^MOD_PATH/install")
    source "libpaho-mqtt3c.a"
    @NoRuntime depends embox.compat.posix.semaphore
}
@Build(stage=1,script="$(EXTERNAL_MAKE)")
@BuildArtifactPath(cppflags="-I$(abspath
                $(EXTERNAL_BUILD_DIR))/third_party/mqtt/paho_mqtt_c/libpaho_mqtt3a/install/")
static module libpaho_mqtt3a {
    @AddPrefix("^BUILD/extbld/^MOD_PATH/install")
    source "libpaho-mqtt3a.a"
    @NoRuntime depends embox.compat.posix.semaphore
}
```

Из этого видно, что можно выбирать, какой именно версией библиотеки хочет воспользоваться разработчик: libpaho-mqtt3c.a или libpaho-mqtt3a.a.

Преимуществом такого подхода является то, что API и документация на проект относятся к самому проекту, а не к специализированной платформе. Кроме того, нужно отметить, что разработка под обычную Linux-систему существенно проще и занимает гораздо меньше времени чем разработка сразу под специфичную платформу.

Ростислав Шаниязов
Москва, Московский институт электроники и математики им. А. Н. Тихонова НИУ ВШЭ

`https://github.com/protocolbuffers/protobuf.git`,
`https://github.com/grpc`

Эффективное взаимодействие: протокол `GRPC` и язык `protobuf`

Аннотация

В современной микросервисной архитектуре остро стоит вопрос об описании жёсткого контракта взаимодействия между сервисами. Язык protobuf позволяет описывать сущности, а также существуют множество расширений для контроля code style, генерации документации, генерации контроллеров и клиентов для протокола grpc, а также контроль обратной совместимости. В этой работе мы подробнее расскажем о его возможностях и покажем конкретные примеры на protobuf и протокола grpc.

Протокол GRPC является детищем кампании Google и довольно популярен в последнее время. Было много статей о том, как работает протокол, в чём плюсы и минусы. В данной статье мы рассмотрим опыт взаимодействия с данным протоколом, рассмотрим нюансы его использования и поговорим, как данный инструмент позволяет нам поддерживать чистый API.

GRPC хорошо интегрирован со средой разработки и его легко использовать.

Зачем?

В парадигме разработки микросервисов первый шаг начинается с API (API First)[1]. С точки зрения CI\CD, микросервисы представляются для нас в виде чёрной коробочки, которая выполняет некоторый контракт. С другой стороны, не хочется ограничивать сам выбор инструмента разработки будь то сервер на java, сервер на Go, либо сервер на NodeJS.

При этом мы хотим иметь инструменты для написания чистого кода: проверка синтаксиса, использование линтера для проверки кода, контроль внесённых изменений: должна присутствовать проверка обратной совместимости новой версии API по сравнению с предыдущей версией, генерация отчёта, который будет в человекочитаемом формате.

Язык Protobuf

Protobuf спроектирован таким образом, что всегда инициализирует примитивы самостоятельно (даже строка инициализируется по умолчанию "") и объекты, если примитивы необязательны, используйте объекты, рекомендуем использовать классы, предоставленные Google Wrappers, так как в них есть вспомогательные методы для работы с примитивами.

Пример описания:

```
// Это описание простого Message
message Message{
    // Тут примитив в виде id-ка в виде 64-битный целочисленного числа
    // По умолчанию 0
    int64 id = 1;
    // Объект  в виде обёртки 64-битный целочисленного числа
    // Этот объект не обязательно инициализировать
    google.protobuf.Int64Value long = 2;
    // Какая-то обязательная строка
    string some_string = 3;
    // Вот тут объект строка, она может быть не инициализирована,
    // если поле необязательно
    google.protobuf.StringValue string = 4;
    // Передаём таймстемп
    google.protobuf.Timestamp date = 5;
    // ...
}
```

Структура описания поля в сообщении состоит в следующем: `[тип сообщения] {название поля} = (номер поля);` .

Для корректной генерации кода рекомендуется следовать style guide [2]

Линтер proto

Линтер proto файлов позволяет нам контролировать стиль описания кода (camel case, структура enum, обязательные опции для proto и прочее). Данный инструмент позволяет облегчить сам процесс ревью API и автоматизировать процесс отлова глупых опечаток и ошибок в нём.

Реализуется с помощью плагина protolint [3]. Распространяется в виде исполняемого файла, который написан на Go.

```
protolint lint -output_file=protolint_results.txt ваш_файл.proto
```

Результат выполнения команды будет записан в `protolint_results.txt`. Если нет ошибок, то этот файл пуст.

Проверка обратной совместимости

Довольно часто в Rest API возникала такая ситуация: было убрано поле в объекте, связь сервер-клиент сломалась. Многие компании решают данную проблему путём жёсткого ограничения лиц, которые имеют доступ к редактированию API и жёсткого палочного подхода, либо раздутие моделек до невообразимого масштаба.

Proto позволяет нам редактировать модельку сообщения таким образом, чтобы связь сервер-клиент не развалилась. Для того чтобы изменить модельку с сохранением обратной совместимости, нужно использовать ключевое слово *reserved* [4]. Резервировать нужно как название поля, так и его номер. Пример:

```
// Это описание простого Message
message Message{
    reserved 2;
    reserved "long";
    // Тут примитив в виде id-ка в виде 64-битный целочисленного числа
    // По умолчанию 0
    int64 id = 1;
    // Какая-то обязательная строка
    string some_string = 3;
    // Вот тут объект строка, она может быть не инициализирована, чтобы её не
    google.protobuf.StringValue string = 4;
    // Передаём таймстемп
    google.protobuf.Timestamp date = 5;
    // ...
}
```

Проверять автоматически, сохранилась ли обратная совместимость, можно с помощью плагина `protolock` [5]. Также распространяется в виде исполняемого файла, написанного на Go.

```
protolock status
```

Программа анализирует все вложенные proto файлы и на основе своих внутренних правил, соответствующие спецификации gRPC. В случае какого-то несовместимого изменения это отобразится в консоль и консоли будет выполнена команда прерывания.

Для того чтобы записать изменения, нужно выполнить команду:

```
protolock commit
```

Генерация кода и документации

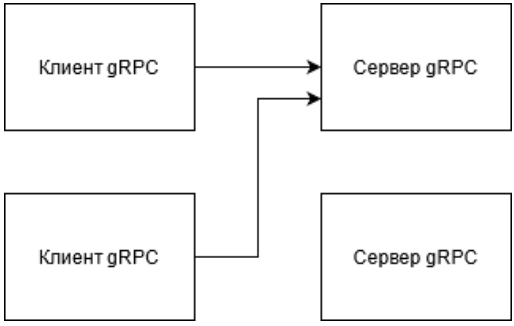
Для генерации кода следует установить golang. И в самом коде установить библиотеку:

```
go get github.com/pseudomuto/protoc-gen-doc/cmd/protoc-gen-doc
```

Потом можно сгенерировать документацию на основе proto файла:

```
protoc -I. -Ithird_party --doc_out=./library-grpc-api-docs/
                                --doc_opt=markdown,file_output.md folder
```

Похожие действия применимы к генерации swagger\кода на Java\Go\Python\C ...



Особенности GRPC

Протокол использует длинные http 2.0 запросы. Соединение открывается и держится около 100 лет, что теоретически позволяет нам

сэкономить время на открытие и закрытие сокета. Следуя из этого, можно выделить следующее:

- Бывают случаи, когда клиент открыл соединение и «умер», не закрыв его. Сервер тратит вычислительные ресурсы на соединение и ждёт, когда клиент либо закроет соединение, либо когда истечёт время соединения (около 100 лет). Для этого нужно принудительно перезагружать сервер. Также от этого спасает Istio в виде Service-Mesh.
- Если вы хотите правильно балансировать нагрузку между клиентами и серверами, нужно использовать внешний балансировщик, иначе можно получить следующую ситуацию: 2 клиента открывают соединение по http к одному серверу и держат его. В итоге получается, что вторая реплика сервера не задействована. Это показано на рисунке.

Литература

[1] Janet Wagner (2021) API-First vs. API Design-First: A Comprehensive Guide. // Stoplight blog (`https://blog.stoplight.io/api-first-vs.-api-design-first-a-comprehensive-guide`) // Просмотрено: 20.04.2022

[2] Google (2022) Protocol Buffers style guide (`https://developers.google.com/protocol-buffers/docs/style`) // Просмотрено: 20.04.2022

[3] Yohei Yoshimuta (2022) protolint (`https://github.com/yoheimuta/protolint`) // Просмотрено: 20.04.2022

[4] Google(2022) Overview | Protocol Buffers | Google Developers (`https://developers.google.com/protocol-buffers/docs/proto3#reserved`) // Просмотрено: 20.04.2022

[5] Steve Manuel (2022) Protolock (`https://github.com/nilslice/protolock`) // Просмотрено: 20.04.2022

И. Е. Панченко
Москва, Postgres Professional
Проект: Postgress Pro

Куда идут слоны

Аннотация
Доклад посвящён современному состоянию PostgreSQL и свежим новостям от разработчиков. Будут освещены наиболее яркие функции предыдущего и ближайшего мажорного релиза, а также более долгосрочные тенденции развития. Посмотрим, каким запросам трудящихся PostgreSQL соответствует, а каким нет и почему. В интерактивном режиме обсудим с залом, чего пользователи ждут от СУБД.

Денисов Е., Александров А. А, Максимов А. Н.
Москва, ЦПР РТСофт

Проблемы безопасности при использовании опенсорс-компонентов в продуктовой разработке. Автоматизация трекинга, зеркалирования и ревью

Аннотация
В докладе будут рассмотрены проблемы использования компонентов с открытым исходным кодом в продуктовых решениях. Проведён анализ рисков, а также представлена оценка достаточности традиционных подходов в современных условиях. Проанализированы возможности использования опенсорс-компоненты для решения задачи построения CI/CD пайплайна для минимизации рисков использования опенсорс-компонентов, а также представлено решение RITMS UP2DATE для безопасного обновления системного и прикладного программного обеспечения встраиваемых систем.

Многие современные программные системы включают те или иные компоненты с открытым исходным кодом. В исследовании [1] показано, что в 2021 году высокий процент отсканированных кодовых баз содержал открытый исходный код. В этом же исследовании указывается, что многие компоненты с исходным кодом могут содержать уязвимости. При использовании опенсорс-компонентов из внешних репозиториев возможно возникновение ситуаций с недоступностью исходного кода или рисков, связанных с несовместимостью лицензий.

В текущих условиях риски использования опенсорс-компонентов необходимо дополнить:

- Возможностью добавления ограничений в лицензии.
- Возможностью ограничения доступа и искажения исходного кода компонентов с открытым исходным кодом, а также репозиториям сторонних опенсорс-пакетов.
- Возможностью запрета доступа к стандартам, базам уязвимостей, и другим источникам информации.

В докладе отмечается, что внедрение практик безопасного программирования и следование рекомендациям [2] недостаточно для предотвращения актуальных рисков в связи с тем, что эти рекомендации нацелены на предотвращение единичных инцидентов. В связи с увеличением числа рисков требуются более согласованные действия опенсорс-комьюнити.

Существует риск, что внесения дискриминационных изменений или ограничений правообладателем компонента с открытым исходным кодом потенциально могут быть осуществлены в любой момент. Возможным решением может быть создание доверенных зеркал репозиториев, в которых будут отслеживаться любые изменения в дереве коммитов.

Рассматривается возможность создания зеркал с минимальными затратами, а также вопросы дальнейшей интеграции их в существующие менеджеры пакетов для различных языков программирования (npm, pip, gopkg, vcpkg).

В качестве примеров в докладе рассматриваются возможные подходы к решению задач автоматизации идентификации и кеширования исходно кода используемых компонентов, а также решение по безопасному обновлению системного и прикладного программного обеспечения на устройствах RITMS UP2DATE.

Таким образом, в докладе рассматриваются методы предотвращения умышленного внесения угроз в программные компоненты, а также практики реального их использования.

Литература

[1] 2022 OPEN SOURCE SECURITY AND RISK ANALYSIS REPORT `https://www.synopsys.com/software-integrity/resources/`

analyst-reports/open-source-security-risk-analysis.html?intcmp=sig-blog-sbom

[2] Security Development Lifecycle (SDL) https://www.microsoft.com/en-us/securityengineering/sdl

Андрей Савченко
Москва, ООО «Базальт СПО»
https://wiki.pine64.org/wiki/PinePhone_Software_Releases

Смартфон на СПО

Аннотация

В докладе делается обзор наиболее популярных смартфонов, работающих исключительно на СПО (в рамках ОС). На примере Pinephone рассматривается опыт повседневной эксплуатации такого смартфона в качестве основного телефона. Рассматриваются основные направления развития СПО для платформы.

Введение

Сейчас почти вся мобильная экосистема захвачена проприетарной дуополией, существенно ограничивающей свободу пользователей на распоряжение собственными устройствами и ПО на них. Однако, за счёт прогресса последних лет, возможность обрести смартфон на СПО всё же есть.

Обзор решений

В данной работе идёт речь именно о возможности использовать смартфон на СПО, с обеспечением основной функциональности без использования проприетарных компонент в операционной системе устройства, за исключением прошивок периферийных устройств (аналогично СПО на десктопах). Просто запустить какое-либо СПО можно на любом устройстве, но это не обеспечит пользователю основные свободы в отношении устройства в целом.

Если не рассматривать исторически значимые, но заброшенные проекты (например, Openmoko) и не рассматривать многочисленные

решения, где пространство пользователя Linux работает поверх проприетарного ядра от производителя (например, Volla), то остаются следующие варианты:

- Replicant [1] — свободная версия Android. Поддерживается несколько широко распространённых устройств, но нужно делать выбор между закрытыми драйверами и/или приложениями и основной функциональностью телефона (звонки, gps, сенсоры и т.п.). Кроме того, ОС телефона плохо изолирована от ОС модема, загрузчика и иных закрытых компонент, что создаёт угрозы безопасности.
- Librem 5 — свободный телефон с возможностью избирательного аппаратного отключения приёмо-передающих устройств и сенсоров. Построен на базе СПО PureOS [2] (производная Debian), можно использовать другие дистрибутивы Linux.
- Pinephone — тоже свободный телефон с аппаратными киллсвитчами; в целом оборудование проще, но и доступнее. Есть широкий выбор дистрибутивов Linux [3] для использования.

Существуют и иные решения, но они либо не обеспечивают базовую функциональность телефона (звонки, sms, навигация), как, например, Nemo mobile или LuneOS, либо содержат закрытые компоненты в базовой системе (например, графический интерфейс), как, например, Maemo, Sailfish OS и её отечественная производная Аврора, что не позволяет рассматривать данные решения как СПО.

Далее детали будут рассмотрены на примере Pinephone, ввиду его практической эксплуатации автором в качестве основного телефона почти год, но большинство результатов будет применимо и к Librem 5 ввиду существенного пересечения кодовой базы.

Программное обеспечение

Дистрибутивы

Основные дистрибутивы являются форками популярных дистрибутивов Linux:

- Mobian (Debian)
- Manjaro (Arch)
- PostmarketOS (Alpine)

- UBPorts (Ubuntu)
- Arch (отдельная сборка)

Необходимость форков вызвана потребностью в дополнительных патчах или фиксации версий ПО, несовместимых с основными ветками. Перечень неполный, т.к. можно использовать любой дистрибутив под Aarch64, если там будет собран нужный софт, включая патчи.

Оболочки

Основные графические оболочки:

- Phosh (Phone shell, GTK)
- Plasma Mobile (KDE)
- SXMO/SWMO (очень самобытная вещь)

Здесь всё сводится к пользовательским предпочтениям: кому-то больше GTK нравится, кому-то KDE, кто-то хочет минималистичный интерфейс. Обычно в дистрибутивах все три варианта сборок доступны, но переключаться на лету между ними просто так нельзя.

Телефония

Основная сложность СПО в мобильных ОС — функции телефонии и мобильной связи, корректное и полнофункциональное взаимодействие с модемом (звонки, sms, mms, передача данных, gnss). Есть два решения:

- ModemManager
- oFono

На данный момент ModemManager стабильнее работает и активнее развивается. Phosh изначально работал на ModemManager, Plasma Mobile перешёл на него недавно с релиза 21.12.

Прикладное ПО

Набор прикладного ПО сильно зависит от дистрибутива и предпочтений пользователя. В целом будет работать любое Linux-приложение, но не все они полностью оптимизированы под мобильный интерфейс. Например, основная функциональность может работать нормально, а какое-либо редкое меню не умещаться на экране — это можно исправить коэффициентом масштабирования, но не очень удобно.

Для желающих запускать Andoid приложения есть Anbox. Здесь примерно как с Wine под Linux: что-то работает хорошо, что-то не очень или вовсе не работает.

В целом основная функциональность (карты, фото, музыка) обеспечены на должном уровне (pure maps, gnome maps, megapixels, lollypop).

Периферия

Пайнфон интересен не только свободной ОС без блобов, но и открытым кодом для существенной части периферии:

- GPU Mali — Lima
- Загрузчик и инициализация памяти — u-boot
- Свободная реализация TF-A (доверенное firmware)
- SCP (system control chip) — базовый управляющий компонент SoC свободен — Crust, а сам чип AR-100 выполнен на базе OpenRISC.

Модем

Пайнфон использует модем Qualcom EG-25, который подключён под USB как внешнее устройство (с точки зрения ОС телефона). На модеме тоже работает Linux на armv7! Из коробки там закрытое решение Qualcom, но исходники свободных компонент были открыты по запросу и сообщество реализовало свободную прошивку [4] с использованием Yocto.

Это позволило, как решить ряд проблем (например, спонтанные зависания или перезагрузки модема), так и расширить функциональность: теперь модему можно отправлять sms, получать dmesg, отложенный звонок и даже будильник; добавлен синтезатор речи для ответных звонков от модема.

При этом всё пространство пользователя модема работает на СПО, а в ядре оставлены два блоба: для trustzone (он устраним, но пока не в приоритете) и для ADSP модема.

ADSP модем представляет собой небольшую real-time OS для Hexagon VLIW и отвечает за работу с полосами частот оператора и sim картой. Его замена на данный момент не реалистична.

Итого, в пайнфоне остаётся всего 2.5 несвободных прошивки в периферии: wifi/bluetooth, ADSP модема и неработающий автофокус задней камеры (заменяется программным фокусом).

Заключение

Итак, смартфон на СПО вполне пригоден к повседневной практической эксплуатации, а высокая ремонтопригодность и открытость схемотехнических решений предоставляет широкие перспективы развития.

Литература

[1] Свободный вариант Android `https://replicant.us`

[2] Репозиторий PureOS `https://repo.pureos.net/pureos/pool/main/`

[3] Дистрибутивы Linux для Pinephone `https://wiki.pine64.org/wiki/PinePhone_Software_Releases`

[4] Открытый SDK для Pinephone (EG25-G) `https://github.com/Biktorgj/pinephone_modem_sdk`

Кирилл Чувилин
Москва, ООО «Открытая мобильная платформа»

Проект: Примеры прикладного ПО для ОС Аврора
`https://community.omprussia.ru/open_source`

Проекты с открытым исходным кодом как основа передачи компетенций разработчикам приложений для ОС Аврора

Аннотация

Компания «Открытая мобильная платформа» является разработчиком российской мобильной ОС Аврора. Как поставщик технологии, мы плотно взаимодействуем с партнёрами, предоставляя необходимые инструменты, документацию, а также нашу экспертизу по разработке прикладного ПО. Важным аспектом передачи опыта и навыков является обмен исходным кодом с техническими специалистами партнёров.

Наиболее удобным форматом такого обмена стала подготовка приложений с открытым исходным кодом, использующих разнообразное API. Доклад посвящён возможностям для компаний, которые разрабатывают прикладное ПО для ОС Аврора, а также задачам, которые возникают при планировании, разработке и публикации приложений, которые используются в качестве примеров.

Компания «Открытая мобильная платформа» (ОМП) [4] является разработчиком средств, позволяющих выстроить доверенную инфраструктуру для работы с мобильными устройствами. Функционирование такой инфраструктуры обеспечивается в том числе операционной системой Аврора и платформой управления устройствами Аврора Центр.

ОС Аврора [1] — это российский POSIX-совместимый дистрибутив Linux, предназначенный для использования на смартфонах и планшетах корпоративными заказчиками и государственными компаниями. Он предоставляет средства защиты информации, что подтверждается сертификатами ФСТЭК и ФСБ [2].

Экосистема приложений ОС Аврора [2] обеспечивается как стандартными приложениями, доступными на устройствах «из коробки» (голосовые вызовы, сообщения, браузер, заметки и т. п.), так и приложениями, разрабатываемым третьими лицами: продуктами партнёров компании или специализированным программным обеспечением (ПО).

Работа ОМП на корпоративном рынке подразумевает регулярное участие в проектах. А это, в свою очередь, означает большое количество специализированного ПО и зачастую — сжатые сроки разработки сложных индустриальных решений.

Со стороны ОМП предусмотрен ряд мер, нацеленных на обеспечение качественной и быстрой разработки приложений партнёрами. Он включает поставку инструментов разработки и документации, организацию технической поддержки и консалтинга по разработке, а также проведение учебных мероприятий. Серьёзным подспорьем в каждом из указанных направлений являются примеры ПО с открытым исходным кодом, которые демонстрируют различные аспекты разработки.

Необходимость в таких примерах обусловлена спецификой ОС Аврора. Основным средством разработки прикладного ПО является Qt [8]. Это зрелый и хорошо документированный фреймворк. Также ОС Аврора предоставляет POSIX-совместимые интерфейсы [7],

которые хорошо знакомы разработчикам приложений для дистрибутивов Linux. Однако для возможности взаимодействия стороннего ПО с функциями мобильного устройства и инфраструктурой, включающей PUSH-уведомления и магазин приложений, в ОС Аврора реализован ряд собственных API. Кроме того, для обеспечения защиты корпоративных данных предусмотрены дополнительные средства безопасности: подпись и валидация установочных пакетов, изоляция исполнения ПО и система разрешений.

Таким образом, наряду со стандартными средствами разработки ОС Аврора предлагает обширный набор собственных технологий. Поэтому, несмотря на наличие документации, для разработчиков не доступны, например, варианты поиска готовых решений в больших публичных базах [7]. Кроме того практика общения с сообществом разработчиков и технической поддержки партнёров показывает, что наглядная демонстрация реализации конкретных функций и подходов, а зачастую и интеграции набора решений, крайне востребована.

Поэтому одной из важных активностей отдела развития и поддержки разработчиков ОМП является подготовка примеров приложений. Вариантами таких примеров являются как небольшие демо отдельных технологий или аспектов разработки, так и законченные решения, которые показывают лучшие практики использования API и особенности их взаимодействия. Но в каждом случае важно уметь решать несколько вопросов:

- какие примеры наиболее востребованы;
- как поставлять;
- на каких условиях предоставлять доступ;
- какие дать возможности по использованию исходного кода;
- как обеспечить поддержку актуальности и развитие.

Доклад посвящён обзору возможных вариантов, а также решений, которые выбраны и используются в ОМП.

Литература

[1] Аврора — первая российская мобильная операционная система, url `https://auroraos.ru`, дата посещения 19.04.2022

[2] Сертификаты, патенты и лицензии ОС Аврора, url `https://auroraos.ru/certificates`, дата посещения 19.04.2022

[3] Каталог приложения для ОС Аврора, url `https://auroraos.ru/applications`, дата посещения 19.04.2022

[4] Открытая мобильная платформа — официальный сайт, url `https://www.omp.ru`, дата посещения 19.04.2022

[5] PosixP1003.1 - Standard for Information Technology–Portable Operating System Interface (POSIX(TM)) Base Specifications, Issue 8, url `https://standards.ieee.org/project/1003_1.html`, дата посещения 19.04.2022

[6] Qt | Cross-platform software development for embedded & desktop, url `https://qt.io`, дата посещения 19.04.2022

[7] Stack Overflow - Where Developers Learn, Share, & Build Careers, url `https://stackoverflow.com`, дата посещения 19.04.2022

Алексей Федченко, Кирилл Чувилин
Москва, ООО «Открытая мобильная платформа»

Проект: Tiny PDF Viewer `https://community.omprussia.ru/open_source`

Использование PDFium совместно с Qt Quick для отображения PDF-документов в ОС Аврора

Аннотация

В докладе описывается реализация библиотеки и приложения для просмотра PDF-документов. Рассказывается про взаимодействие со структурой документа (страницы, текст, аннотации, заметки), а также особенности рендеринга контента (в том числе, ограничения видеобуфера, отрисовка тайлами и масштабирование). Полученное решение основано на технологиях Qt Quick и PDFium. Оно может быть удобно использовано в приложениях, написанных с использованием фреймворка Qt, в том числе для ОС Аврора.

ОС Аврора [1] — это российский POSIX-совместимый дистрибутив Linux, предназначенный для использования на смартфонах и планшетах корпоративными заказчиками и государственными компаниями. Экосистема приложений ОС Аврора [2] обеспечивается как стандартными приложениями, доступными на устройствах «из коробки» (голосовые вызовы, сообщения, браузер, заметки и т. п.), так и приложениями, разрабатываемым третьими лицами: продуктами партнёров компании или специализированным программным обеспечением (ПО).

Одной из наиболее востребованных функции корпоративного ПО является работа с документами в формате PDF: просмотр, аннотирование, переход по ссылкам. Это порождает задачи, требующие интеграции функций для работы с PDF в другие приложения: системы электронного документооборота, подпись документов, платёжные системы и др.

Поэтому возникает вопрос о том, какие доступны средства разработки, позволяющие взаимодействовать с документами в формате PDF [14, 15, 3, 4, 5]. Технически могут быть использованы любые POSIX-совместимые решения [7, 6]. Однако условия их применения на мобильных устройствах, которые уступают в производительности ПК, накладывают требования к скорости работы. А потребность использования в сторонних приложениях обуславливает необходимость пермиссивной лицензии.

В статье [17] описан выбор библиотеки для работы с PDF, которая может быть использована для таких целей с учётом всех требований. Наиболее подходящей оказалась PDFium [16] — библиотека с открытым исходным кодом от Google, распространяющаяся под лицензией MIT [10]. Решение, описываемое в докладе, основано именно на ней.

Основным фреймворком для разработки прикладного ПО для ОС Аврора является Qt версии 5.6 [8], а интерфейс пользователя реализуется с помощью QML [12]. Это означает, что для удобства использования в приложениях нужно обеспечить совместимость PDFium и Qt. Есть решения, которые реализуют подобное связывание:

- Библиотека QtPDF [13] предоставляется разработчиками Qt, но её лицензия GPLv3 [9] требует раскрытия исходного кода конечного приложения, что делает её неподходящей для большинства прикладных коммерческих приложений.
- QtPDFium [18] — это неофициальная библиотека. Она распространяется под удобной лицензией BSD [11], но при этом использует непубличный API и привязана к устаревшей версии PDFium. Кроме того, поддержка библиотеки прекращена.

Таким образом, для свободного использования актуальной версии PDFium совместно с Qt потребовалось разработать новое связующее ПО.

При реализации связующего ПО и приложения его использующего, возникает ряд задач, обусловленных как функциональными требованиями, так и техническими ограничениями:

- как обеспечить асинхронную обработку данных, которую активно использует Qt Quick, при условии строгой однопоточности PDFium;
- как отрендерить и отрисовывать страницу, разрешение которой превышает размер видеобуфера устройства;
- как реализовать представление списка страниц с корректной прокруткой и эффективным вычислением текущего положения при условии, что страницы могут быть разных пропорций;
- как масштабировать одну страницу и список страниц;
- как сделать взаимодействие пользователя с гиперссылками и другими элементами навигации;
- каким образом реализовать выделение и копирование текстовых данных;
- как организовать работу с заметками в документе.

В докладе описываются выбранные подходы для решения указанных задач на примере библиотеки AmberPDF и приложения Tiny PDF Viewer, которые были реализованы и предоставляются для удобства работы с PDF в приложениях для ОС Аврора, но также могут быть полезны и при разработке других приложений, основанных на Qt и работающих с PDF.

Литература

[1] Aurora OS website, url `https://auroraos.ru/`, дата посещения 12.03.2021

[2] Каталог приложения для ОС Аврора, url `https://auroraos.ru/applications`, дата посещения 12.03.2021

[3] Document management — Portable document format — Part 1: PDF 1.7, url `https://www.adobe.com/content/dam/acom/en/devnet/acrobat/pdfs/PDF32000_2008.pdf`, дата посещения 12.03.2021

[4] Portable Document Format: An Introduction for Programmers, Kas Thomas, MacTech Magazine, v15, № 9, 1999 г.

[5] Dejan Lukan, Basic Structure [updated 2020] – Infosec Resources, url `https://resources.infosecinstitute.com/topic/pdf-file-format-basic-structure/`, дата посещения 12.03.2021

[6] List of PDF software — Wikipedia, url https://en.wikipedia.org/wiki/List_of_PDF_software#Linux_and_Unix, дата посещения 12.03.2021

[7] PosixP1003.1 — Standard for Information Technology — Portable Operating System Interface (POSIX(TM)) Base Specifications, Issue 8, url https://standards.ieee.org/project/1003_1.html, дата посещения 12.03.2021

[8] Qt | Cross-platform software development for embedded & desktop, url https://qt.io, дата посещения 12.03.2021

[9] The GNU General Public License v3.0 - GNU Project - Free Software Foundation, url http://www.gnu.org/licenses/gpl-3.0.html, дата посещения 12.03.2021

[10] MIT License Definition, url http://www.linfo.org/mitlicense.html, дата посещения 12.03.2021

[11] BSD License Definition, url http://www.linfo.org/bsdlicense.html, дата посещения 12.03.2021

[12] QML Applications | Qt 5.15, url https://doc.qt.io/qt-5/qmlapplications.html, дата посещения 12.03.2021

[13] Qt PDF | Qt Marketplace, url https://marketplace.qt.io/products/qtpdf, дата посещения 12.03.2021

[14] PDFISO — ISO 32000-1:2008 — Document management — Portable document format — Part 1: PDF 1.7, url https://www.iso.org/standard/51502.html, дата посещения 12.03.2021

[15] Betsy Fanning, PDF Standards....transitioning the PDF specification from a de facto standard to a de jure standard, MacTechAIIM Magazine, July/August, 2007

[16] GitHub — chromium/pdfium: The PDF library used by the Chromium project, url https://github.com/chromium/pdfium, дата посещения 12.03.2021

[17] Alexey Fedchenko and Kirill Chuvilin, PDF Document Rendering on Mobile Devices in the Case of Aurora OS, book29th Conference of Open Innovations Association, p. 118–124, 2021, FRUCT

[18] GitHub | qtpdfium, url https://github.com/paulovap/qtpdfium, дата посещения 2022-03-12

Александр Епифанов, Дмитрий Солдатенков
Санкт-Петербург, Tau Технологии

Проект: RhoMobile `https://github.com/rhomobile/rhodes`, `https://github.com/rhomobile/rhoconnect`

Разработка приложений для мобильных и настольных платформ с помощью единого инструмента

Аннотация

В докладе будет рассказано об открытом фреймворке RhoMobile, позволяющим создавать переносимые приложения для большинства актуальных на сегодняшний день мобильных и настольных платформ на языках Ruby и JavaScript. Список целевых платформ фреймворка включает мобильные: iOS, Android, WinCE, Sailfish/Аврора и настольные: Windows, OS X, различные дистрибутивы Linux (Ubuntu, ALT, ROSA, RED, Astra). Будут освещены особенности архитектуры фреймворка, способы использования, а также продемонстрированы примеры создания и работы приложений на базе фреймворка.

Описание и назначение

RhoMobile Suite — инструмент для создания переносимых мобильных и настольных приложений. Доступен под лицензией MIT.

Приложение на RhoMobile называется нативно-гибридным. Код приложений похож на код веб-приложений. Пользовательский интерфейс в основном пишется на HTML5, CSS3, а логика на JavaScript, либо Ruby.

История с 2008 по 2022

2008–2011 — стартап.

2011–2014 – часть Motorola Solutions.

2014–2015 — часть Zebra Technologies.

До 2015 года разработку вела команда инженеров из России, США и Великобритании.

Текущий статус

С 2015 года часть оригинального коллектива из России осуществляет дальнейшее развитие и поддержку проекта в составе Tau.

За последнее время была добавлена поддержка Sailfish OS, и дистрибутивов Linux, в т.ч. Alt, ROSA, Red OS, Astra.

Стабильная версия — 7.4.2, бета версия — 7.5.0.beta2.

Архитектуры мобильных приложений

- Классическое нативное приложение. Нативные инструменты iOS, Android.
- Кроссплатформенное нативное приложение — Xamarin, React Native, QT, Flutter и др.
- Веб приложение — «сайт», запущенный в системном браузере.
- Гибридное приложение — веб-приложение, обёрнутое в нативный контейнер. Apache Cordova/PhoneGap, Rhomobile (в чисто гибридном режиме).
- Смешанно-гибридное приложение — совмещение нативного, кроссплатформенного и гибридного подходов. Интерфейс реализуется на HTML/CSS, логика пишется на переносимом инструменте и исполняется отдельно от веб-компонента. Пример — RhoMobile.

Особенности смешанно-гибридного подхода

Локальный сервер

Разделение UI и логики приложения осуществляется с помощью HTTP(S) сервера внутри процесса приложения.

Веб-компонент обращается серверу, он исполняет логику и выдаёт ответ в виде HTML.

Веб-компонент также может вызывать API напрямую из JS.

Ruby

В качестве средства реализации логики выбран язык Ruby. Для клиент-серверного взаимодействия используется адаптированный фреймворк Ruby-on-Rails.

Архитектура может быть реализована и с помощью других языков. В частности, в RhoMobile, помимо Ruby, можно также выбрать Node.js.

Расширения

В RhoMobile используется механизм CommonAPI, позволяющий описать модули, методы и свойства в общем виде на XML.

Сборка в виде библиотеки

RhoMobile имеет собственную систему сборки. В некоторых случаях разработчикам может потребоваться встроить часть функций платформы в их нативный проект. Для этого существует режим сборки платформы в виде библиотеки для целевой платформы.

Производительность

Производительность интерпретирующей машины: код на Ruby при сборке приложения компилируется в бинарный байт-код iseq.

Механизм расширений позволяет вынести ресурсоёмкие операции в нативную часть.

Часть API возможно вызывать в асинхронном режиме.

Современные веб-движки хорошо оптимизированы.

Нативный «look&feel»

Для реализации интерфейса можно выбрать frontend фреймворк, эмулирующий визуальные компоненты конкретной платформы.

Вопросы безопасности

Для защиты веб-сервера используется двусторонняя SSL аутентификация с генерацией ключей и сертификатов при каждом запуске приложения.

Поддерживаемые платформы

Целевые: iOS, Android, WinCE, Windows, Sailfish, Linux (Ubuntu, ALT, RED, ROSA, Astra).

Хостовые: Windows, OS X, Linux.

Функции, связанные с применением в бизнес-приложениях

Защита данных

- шифрование локальной БД;
- шифрование файлов кода и данных в установочном пакете приложения;
- детектирование запуска приложения на эмуляторе, под отладчиком, либо на взломанном устройстве.
- на Android: проверка сигнатуры подписи пакета, проверка идентификатора установщика.

Бесшовная интеграция

Применяется клиентская библиотека и промежуточный сервер RhoConnect. Они обеспечивают синхронизацию моделей данных приложения и контроль конфликтов.

Поддержка спецустройств

API платформы поддерживает функции ТСД[1], такие как сканирование штрихкодов, чтение RFID меток, беспроводную Bluetooth печать и др.

Состав проекта

- Rhodes — фреймворк и набор инструментов для разработки приложений.
- RhoConnect — сервер интеграции данных.
- RhoStudio/ VSCode debugger — IDE для редактирования и отладки приложений.

[1] Терминал Сбора Данных

Продукты на базе RhoMobile

Браузер, приложения для мобильного сервиса, банкинг, страхование, энергетика и др.

В завершающей части доклада будет продемонстрировано создание и запуск приложения на RhoMobile.

Кулагин Владимир Петрович, Муравьёв Николай Дмитриевич
Москва, МИРЭА
https://github.com/tuplecats/GPTN

Использование свободного ПО для разработки средств моделирования сетевых моделей сложных систем

Аннотация

В докладе рассматриваются вопросы использования свободного программного обеспечения для создания пакета программ, предназначенного для исследования сетевых моделей сложных параллельных систем. В качестве инструмента для построения сетевых моделей используется аппарат сетей Петри. Создаваемый пакет программ также относится к свободному программному обеспечению.

Понятие сложной дискретной динамической системы является одним из наиболее общих понятий в информатике и вычислительной технике. В настоящее время дискретными сложными системами являются компьютеры, вычислительные сети, специализированные процессоры различной структуры, системы обработки данных, сенсорные сети и др.

Одним из инструментов для решения задач, связанных с исследованием сложных систем, является аппарат сетей Петри. Существуют подходы для поиска альтернативных структур сложных систем с использованием сетей Петри, они заключаются в декомпозиции и синтезе. Благодаря таким подходам построения вариантов на основе обобщённой модели позволяет проектировщику сложной системы получить множество возможных альтернативных структур, из которых выбрать оптимальные структуры, применив методы метрической

оценки. Однако такой подход является NP-сложной задачей, для решения которой необходимы вычислительные ресурсы.

Сеть Петри (СП) — математический аппарат для формализации, анализа и моделирования дискретно-событийных систем. СП может использоваться для эффективного моделирования параллельных распределённых систем, а также технологических процессов.

Программный

Мотивация разработки программного комплекса для анализа и моделирования параллельных процессов на основе сетевых моделей опирается на наличие малого количества и неразвитость существующего программного обеспечения в данном направлении. Текущие решения предлагают следующие возможности [4]:

- задание СП-моделей;
- анализ СП-моделей с помощью имитации поведения сетей Петри;
- построение и анализ пространства состояний модели (построение дерева достижимых разметок);
- оценка сетей на живость, ограниченность, наличие конфликтов.

Разрабатываемый комплекс программ включает функционал существующих программных средств, а также содержит новые компоненты.

Программный комплекс написан на двух языках программирования: C++ и Rust и состоит из следующих модулей и составляющих:

- графический интерфейс;
- модуль визуализации сетей Петри;
- модуль декомпозиции;
- модуль синтеза новых структур;
- модуль построения дерева достижимых разметок;
- модуль анализа СП на основе решения систем линейных уравнений.

Графический интерфейс служит для возможности интерактивной работы с СП-моделями, т. е. является frontend для остальных модулей. Для написания графического интерфейса используется фреймворк Qt [6]. Qt является кроссплатформенным и позволяет писать

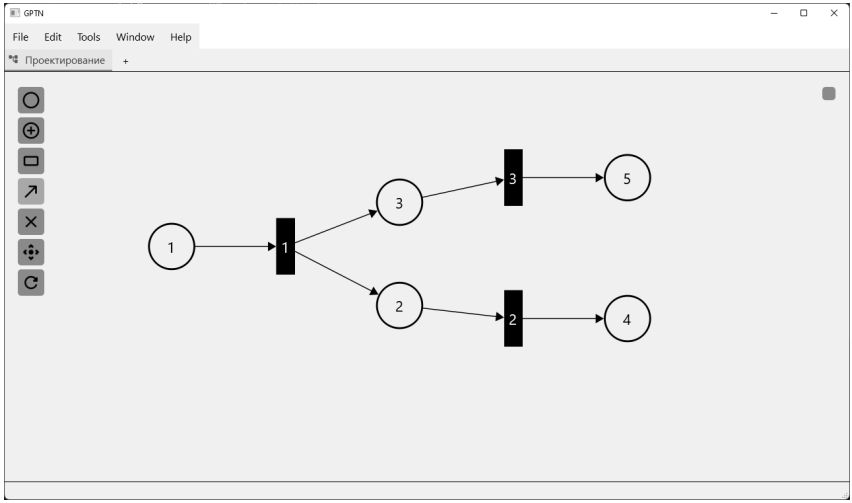


Рис. 1: Визуализация сети Петри алгоритмом Dot.

программное обеспечение под разные семейства ОС, а также позволяет писать свободное ПО используя лицензии GNU GPLv3 (GNU General Public License, version 3) или GNU LGPLv2.1 (GNU Lesser General Public License, version 2.1).

В качестве модуля визуализации используется ПО Graphviz [7], которое позволяет визуализировать графы, а в данном случае сети Петри, в удобном виде посредством алгоритмов визуализации. Graphviz распространяется по свободной лицензии EPL (Eclipse Public License v1.0). На данный момент программный комплекс использует 3 алгоритма: dot, FDP (Force Directed Placement), SFDP (Scalable FDP). Другие алгоритмы, такие как patchwork, circo, osage, neato поддерживаются, но не используются. Пример использования ПО Graphviz в разрабатываемом программном комплексе изображён на Рис. 1.

Модуль декомпозиции производит разделение сети на линейно-циклические и линейные базовые фрагменты для дальнейшего их использования в модуле синтеза новых структур. Алгоритм декомпозиции основан на преобразовании матрицы инцидентности сети Петри [2]. Для реализации алгоритмов использовалась библиотека

openBLAS [5], которая позволяет эффективно работать с матрицами и линейной алгеброй. OpenBLAS представляет собой реализацию BLAS (базовые подпрограммы линейной алгебры) и LAPACK API интерфейсов со множеством оптимизаций для разных типов процессоров. OpenBLAS распространяется по лицензии «BSD 3-Clause».

Модуль синтеза новых структур является прямым продолжением модуля декомпозиции. Данный компонент также использует библиотеку openBLAS. Основной задачей синтеза является поиск всех возможных структур, которые можно получить, опираясь на исходную сеть Петри.

Функционал программного комплекса активно развивается и расширяется, что позволит в будущем поддерживать разный набор методов анализа сетей Петри. В краткосрочном плане стоит решение задачи синтеза новых структур сетей Петри при сохранении первоначально заданных функций, но обладающих более эффективными показателями. Эффективное решение этой задачи возможно с использованием вычислений на GPU.

Используемые сторонние библиотеки и фреймворки в работе имеют разные лицензии. Для лицензирования разрабатываемого программного комплекса используется лицензия LGPLv2.1.

Литература

[1] Структурный анализ сетей Петри. / В. П. Кулагин, В. Н. Дубинин // Информационные технологии. — 2016 — Т. 22. — № 2. — с. 3–13.

[2] Тензорные методы исследования структур сетей Петри. / В. П. Кулагин // Информационные технологии. — 2015. — Т. 21. — № 2. — с. 83–94.

[3] Проектирование матричных вычислительных структур с использованием сетей Петри. / В. П. Кулагин, Е. С. Малых // Информационные технологии. — 2019. — Т. 25. — № 5. — С. 271–283.

[4] Анализ программных средств для работы с сетями Петри / В. П. Кулагин, А. А. Логинов // Информационные технологии. — 2021. — Т. 27. — № 2. — С. 89–96.

[5] OpenBLAS: An optimized BLAS library. URL: `https://www.openblas.net`

[6] Qt Documentation Home. URL: `https://doc.qt.io`

[7] Graphviz. URL: `https://graphviz.org`

Костарев Алексей Федорович
Пермь, ООО «Базальт СПО»
https://packages.altlinux.org

ALT Container OS — аналог Fedora CoreOS на пакетной базе ALT Linux

Аннотация

В докладе рассматривается текущая реализация ALTCOS — аналога Fedora CoreOS — минимальная операционная система с автоматическим обновлением, предназначенная для запуска задач в контейнерах безопасно и с масштабированием

ALT Container OS (ALTCOS) — дистрибутив на основе пакетной базы ALT Linux, являющий аналогом Fedora CoreOS, имеющий следующие особенности:

- минимальный набор пакетов для поддержки контейнеризации (docker, docker-compose, docker swarm, podman);
- малый объём, занимаемый на диске $\approx$ 1.2GB;
- минимальное время (пере)загрузки системы — несколько секунд;
- повышенная защищённость за счёт монтирования в режиме *только на чтение* системных каталогов;
- монолитность — атомарные обновления с возможностью отката на предыдущую версию;
- поддержка потоков, для различных платформ и архитектур ALT Linux. Поддерживаются потоки `altcos/x86_64/sisyphus`, `altcos/x86_64/p10` (архитектура x86_64, платформы sisyphus, p10). В дальнейшем планируется добавить поддержку потоков для других платформ и архитектур;
- автоматическое развёртывание на множестве (виртуальных) машин без участия оператора;
- поддержка различных режимов автоматического обновления дистрибутива без участия оператора, в том числе режима с согласованной перезагрузкой узлов кластера, при которой кластер сохраняет свою работоспособность.

Обновления ALTCOS производятся с помощью `ostree`, обеспечивающего обновление системы целиком за одно действие. При необходимости есть возможность откатить состояние системы в предыдущее состояние.

Во время разворачивания настройки системы производится с помощью конфигурационных файлов `ignition`. Данный механизм позволяет без участия оператора гибко настраивать параметры системы (размеры и типы файловых систем, сетевые настройки, запускаемые сервисы и т.п.). Кроме того, использование механизма `ignition` для разворачивания узлов кластера позволяет в дальнейшем использовать ALTCOS для разворачивания решений на базе openstack.

ALTCOS-образы в формате ISO и QCOW2 доступны на сайте `https://altcos.altlinux.org/`.

В настоящее время для архитектуры x86_64 поддерживаются следующие платформы:

- sisyphus — поток `altcos/x86_64/sisyphus`;
- p10 — поток `altcos/x86_64/p10`;
- последующие платформы.

В дальнейшем предполагается поддержка других архитектур.

Кроме того, поддерживается механизм создания в рамках каждого потока подветок со специализированным программным обеспечением. Например: `altcos/x86_64/Sisyphus/k8s` — подветка с kubernetes и т.п.

ALT Container OS поддерживает два основных режима работы с контейнерами:

- серверный — демон dockerd с клиентским приложением docker;
- бессерверный — клиентское приложение podman.

Серверный вариант позволяет клиентским приложениям (docker и др.) работать с демоном dockerd как локально, так и удалённо через REST-интерфейс, но является менее защищённым, чем бессерверный вариант podman.

Бессерверный вариант podman, появившийся относительно недавно, выполняет функции сервера и работает напрямую с файловой системой.

Оба клиентских приложения имеют похожий набор команд, но хранят образы и контейнеры в различных каталогах. Образы, ска-

чанные приложением docker, недоступны приложению podman и наоборот.

Серверный вариант (docker, dockerd) удобен для проектов с небольшим числом запускаемых сервисов (не более десятка).

Он позволяет запускать как отдельные контейнеры, так и по YML-файлу описания стеки сервисов как на одном сервере (через docker-compose), так и на кластере серверов (docker stack).

Бессерверный вариант (podman) позволяет запускать отдельные контейнеры и используется как основной в кластере kubernetes.

Реализована возможность создания зеркал центрального репозитория `https://altcos.altlinux.org/` с механизмом создания дополнительных веток содержащих пакеты, необходимые для функционирования узлов. Это позволит уменьшить объём трафика и сократить время разворачивания и обновления ALTCOS-кластера.

Литература

[1] ALT Container OS: `https://www.altlinux.org/ALT_Container_OS`

[2] ALT Container OS подветка K8S: `https://www.altlinux.org/ALT_Container_OS_подветка_K8S`

[3] ALT Container OS подветка K8S. Создание HA кластера: `https://www.altlinux.org/ALT_Container_OS_подветка_K8S.Создание_HA_кластера`

Данила Загайнов, Георгий Курячий
г. Москва, ВМК МГУ им. М. В. Ломоносова, ООО «Базальт СПО»

Проект: rpm-build-python3
`http://git.altlinux.org/gears/r/rpm-build-python3.git`

Проблема python3-зависимостей в пакетах в Sisyphus

Аннотация

В Sisyphus для работы с пакетами, имеющими python3 зависимости, а также предоставляющими python3 provides, используется rpm-build-python3.

Этот пакет потребовал серьёзной доработки и вот что уже было сделано:

- После выхода python 3.10 rpm-build-python3 стал непригоден, так как python3.req.py, его компонент для определения зависимостей, использовал модули parser и symbol, более не поддерживаемые в свежих релизах python3. Поэтому python3.req.py был переписан, используя модуль ast.

Вот что уже готово к внедрению:

- определение зависимостей при использовании importlib
- отключение поиска зависимостей для кодировок

Вот что планируется к внедрению:

- более грамотное определение provides, предоставляемых пакетом
- предоставление возможности использования python3.req.py не только лишь при сборке
- вынесение python3.req.py в отдельный модуль

Определение python 3 зависимостей

Для определения python3 зависимостей в Sisyphus используется python3.req.py, компонент пакета rpm-build-python3. Первоначально для этой процедуры были задействованы модули parser и вспомогательно symbol. В конечном итоге этот процесс сводился к рекурсивному распарсиванию кода по частям вплоть до нахождения искомого символа или до упора.

Однако при разработке rpm-build-python3 0.1.18 и переписывании кода python3.req.py был применён модуль ast, а модули parser и symbol были отброшены. Как следствие, алгоритм обработки кода целевого файла свёлся к прохождению по дереву AST и поиску объектов типа ast.Import, ast.ImportFrom и т.д.. В качестве бонуса python3.req.py стал работать быстрее и пропало порождение одинаковых зависимостей при обработке конструкций __import__(smth).

Приятным последствием переписывания python3.req.py стала возможность включить определения зависимостей при использовании конструкций importlib, а также отключение поиска зависимостей для кодировок. Однако обе опции пока не были внедрены. Внедрение importlib допускает возникновение проблемы при определении зависимостей, создаваемых конструкциями вида:

```
smth = 'os'
importlib.import_module('%s' % smth)
```

Определение self-provides

Однако после определения зависимостей остаётся вопрос, а все ли зависимости могут быть удовлетворены дополнением к подмножеству, называемым в рамках задачи исследуемым пакетом, множества всех пакетов Sisyphus? Сам пакет может предоставлять свои собственные сабмодули и даже целый модуль, то есть его self-provides должны быть исключены из списка его зависимостей. Для этого получившиеся зависимости проходят постобработку путём фильтрации через список provides, предоставляемых данным пакетом.

Сами же provides формируются из тех файлов, которые попали к python3.req.py.

В случае если пришедший путь к файлу содержит суффикс из sys.path, тогда можно предположить, что это какой-то модуль, который может содержать self-provides и тогда запускается рекурсивный проход по всему модулю. Однако существуют случаи, когда пакет ничего явно не провайдит, а содержит лишь какие-то свои «плагины». Вероятно, что компоненты пакета могут пытаться импортировать такие плагины, однако провайдить их он не собирается. На этот случай в python3.req.py была реализована переменная окружения RPM_PYTHON3_IMPORT_PATH, в которую раскрывается макрос %_python3_import_path. Через этот макрос добавляются дополнительные пути, по котором python3.req.py должен определить provides. Также этот макрос весьма удобен в случаях, когда сборка пакета разбивается на 2 разных подпакета, но при этом компоненты одного импортируют компоненты другого.

Литература

[1] Ссылка на сайт python `https://docs.python.org/3/reference/import.html`

[2] Ссылка на документацию по модулю ast `https://docs.python.org/3/library/ast.html`

Егор Игнатов
Обнинск, ООО «Базальт СПО»

Коммуникация сборочницы со сторонними сервисами через брокер сообщений RabbitMQ

Аннотация

Доклад посвящён интеграции протокола AMQP в сборочницу для отправки сообщений о результатах её работы, а также обработке этих сообщений сторонними сервисами.

Для получения актуальной информации от различных сервисов, задействованных в ALT Linux, было решено создать инфраструктуру обмена сообщениями. Такая система поможет не только узнавать об изменениях, но и автоматизировать разные процессы.

Инфраструктура обмена сообщениями состоит из трёх основных компонентов: *Продюсеров* — те, кто оправляют сообщения, *Брокера* — он принимает и распределяет сообщения по очередям, в данном случае это `RabbitMQ`, и *Консюмеров* — они забирают эти сообщения из очереди брокера и обрабатывают их.

Продюсеры

Основным продюсером является наша сборочница (`girar`), помимо неё сообщения отправляет `bugzilla` и некоторые прочие сервисы. В докладе будут рассмотрены:

- Изменения добавленные в проект `girar` для поддержки отправки AMQP сообщений об изменениях в сборочных заданиях.
- Какие типы сообщений от `girar` бывают и когда они отправляются.
- Как работает отправка сообщений из `bugzilla`.

Брокер сообщений

Отправленные сообщения получает `RabbitMQ`, рассмотрим:

- Конфигурацию и работу сервера `RabbitMQ`.
- Устройство High Availability кластера средствами `RabbitMQ`.
- Роутинг сообщений на стороне сервера

Консюмеры

В данной части доклада мы рассмотрим какие есть сервисы, что они делают и как работают. Как сервисы взаимодействуют с проектами altrepo-api (`https://rdb.altlinux.org/api/`) и `packages.altlinux.org`. А также прочие применения и дальнейшие планы по развитию инфраструктуры.

Литература

[1] RabbitMQ Documentation, `https://www.rabbitmq.com/documentation.html`

Виталий Липатов
Санкт-Петербург, ООО «Базальт СПО»

Проект: EPM https://www.altlinux.org/Epm

Управление сторонними пакетами с помощью EPM

Аннотация

Размещаемые в репозитории пакеты должны иметь свободную лицензию, разрешающие их публикацию. Установка стороннего, проприетарного ПО зачастую осложнена тем, что оно не предназначено для установки в системы на базе ALT. Команда `epm play` позволяет устанавливать такое ПО, скачивая его с сайта производителя и производя конвертацию в rpm-пакет, адаптированный для ALT, с помощью команды epm repack. Также в докладе рассматриваются другие подходы к установке стороннего ПО, поддерживаемые в универсальном пакетном менеджере EPM.

О несовместимости

Как правило, подготовленные сторонними разработчиками пакеты собраны и проверены только для нескольких систем, в число которых, особенно если разработчик зарубежный, ALT не входит. С одной стороны, пользователя надо оградить от возможности сломать систему и нарушить безопасность, установив откуда-то скачанный пакет. С другой стороны, нужно предоставить доступ к популярным или необходимым приложениям.

Перепаковка сторонних пакетов

Для адаптации сторонних пакетов к нормам пакетов ALT в `epm` добавлена команда `repack`, которая перепаковывает пакет, при этом выполняя заранее заданный рецепт, совпадающий с названием пакета, выполняющий все подготовительные действия так, что установочные скрипты становятся не нужны, а все изменения в файловой системе в итоге делаются устанавливаемым пакетом.

Использовать перепаковку можно либо в виде `epm repack` (перепакует пакет и создаст рядом новый), либо в виде `epm install--repack` (установит пакет с перепаковкой на лету).

Важным свойством перепаковки является то, что перепаковывать можно пакеты любых пакетных менеджеров, и даже архивы, а в последних версиях EPM появилась поддержка установки/перепаковки AppImage (формат для распространения переносимых приложений для Linux). Основана перепаковка на использовании `alien` и `rpm-build`.

При разработке решения по перепаковке выяснилось, что даже тарболы с программами можно легко перепаковывать в rpm пакет. Например, Telegram поставляется в виде архива tsetup.tar.gz, который, по задумке авторов, может быть распакован и запущен из любого места. Таким образом, он отлично чувствует себя в каталоге `/opt`.

Задача перепаковки пакета на самом деле является задачей интеграции в систему, поэтому для графических приложений важно исправить упаковку `desktop`-файла, значков к приложению, а серверные программы должны приобрести корректные `unit`-файлы.

epm play

Чтобы приступить к перепаковке, пакет с программой ещё нужно скачать, что не всегда является простым занятием: нужно найти, что скачать, выбрать пакет той системы, что лучше подойдёт для ALT...

Чтобы избавить пользователя от этого выбора, придумана команда `epm play`.

По сути, она предоставляет консольный интерфейс как бы к магазину приложений.

В итоге можно ввести команду `epm play yandex-browser` и Яндекс браузер будет правильным образом установлен в вашу систему.

Реализовано это через набор рецептов для установки в каталоге `/etc/eepm/play.d`: там указывается название пакета, формируется URL для скачивания, в зависимости от архитектуры процессора, версии системы и т.п, производится установка пакета.

Простейший рецепт выглядит так:

```
#!/bin/sh
PKGNAME=viber
DESCRIPTION="Viber for Linux from the official site"
. $(dirname $0)/common.sh
[ "$($DISTRVENDOR -a)" != "x86_64" ] && echo "Only x86_64 is supported" && exit 1
epm install "https://download.cdn.viber.com/cdn/desktop/Linux/viber.deb"
```

Скачивание файлов по маске

В основном установка в скриптах для `epm play` основана на способности `epm` устанавливать пакеты по URL, и даже по URL с маской (когда точная версия пакета неизвестна, по wildcard будет выбрана самая новая версия).

Эта способность поддерживается встроенной в epm утилитой eget, которая изначально представляла из себя обёртку вокруг wget, а сейчас работает через curl или wget. Цель в том, чтобы предоставить более свободный доступ к скачиванию по маске.

Например, можно не только написать

```
eget --latest https://download.example.com/packages/package-*.tar.gz
```

или

```
eget --list https://download.example.com/packages/package-*.tar.gz
```

а скачать пакет со страницы, которая не рассчитана на прямую отдачу:

```
eget --latest https://example.com/somepage package-*.tar.gz
```

Обратите внимание на пробел между URL и маской файла — файл будет найден на указанной странице и скачан.

Есть и отдельная поддержка github для скачивания релизов.

Благодаря отделённости этого функционала в `eget`, скрипт для `epm play` выглядят достаточно лаконично.

Перспективы и варианты использования

Проект EPM открыт для расширения, необходимые сценарии можно как присылать в виде pull request на github или иным способом, так и паковать в отдельные пакеты.

Также `epm` и механизм сценариев в нём могут быть применены для создания установочных скриптов. Благодаря тому, что `epm` хорошо умеет определять систему, дистрибутив и всякие подробности (см. ниже вывод команды `epm print info`), можно предусматривать различные нюансы, не изобретая велосипед.

```
$ epm print info
    Pretty distro name (--pretty): ALT Sisyphus Sisyphus (20201124)
     Distro name and version (-e): ALTLinux/Sisyphus
     Package manager/type (-g/-p): apt-rpm / rpm
     Running service manager (-y): systemd
              Virtualization (-i): (host system)
            CPU Cores/MHz (-c/-z): 4 / 3182 MHz
            CPU Architecture (-a): x86_64
     CPU norm register size  (-b): 64
     System memory size (MB) (-m): 5511
                Base OS name (-o): linux
Base distro (vendor) name (-s|-n): alt
```

Сейчас скрипты установки, а тем более перепаковки недостаточно универсальны, но есть интерес и намерение увидеть `epm play` достаточно одинаково работающим и на других популярных системах, поскольку важной целью создания EPM было сглаживание различий между различными дистрибутивами перед лицом пользователя и системного администратора.

Евгений Шестепёров
Обнинск, ООО «Базальт СПО»

Автоматизация процессов в рамках тестирования сборочных заданий для стабильных репозиториев ОС ALT Linux

Аннотация

Доклад посвящён автоматизации процессов тестирования заданий, попадающих в стабильные репозитории, в компании «Базальт СПО». К таким процессам относится создание задач в системе контроля проектов Redmine; генерация сводного отчёта о сборочном задании; автоматическая проверка собираемости зависимых пакетов с заданием;

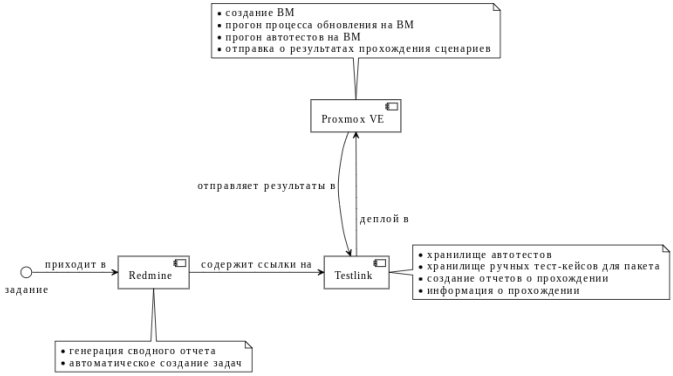


Рис. 1: Упрощенная схема взаимодействия компонентов системы между собой.

разворачивание виртуальных машин в системе виртуализации Proxmox VE для тестирования сборочного задания; определение и загрузка автоматизированных тестов; отправка отчётов о тестировании в систему управления тестами Testlink.

Стабильные ветки репозитория пакетов ALT Linux создаются на основе нестабильного репозитория Sisyphus с целью разделения темпов изменений разработки и эксплуатации путём стабилизации и тестирования. На данный момент активно проверяются продуктовые репозитории p10, p9 и сертифицированные ветки. При тестировании заданий инженеры QA часто выполняют рутинные действия, такие как создание виртуальных машин, проверка пересборки зависимых пакетов с заданием, обнаружение файловых конфликтов, заполнение прогонов по тестовым сценариям. Поэтому одной из первых и важнейших задач для инженеров QA Team стало создание системы контроля над пропуском заданий, которая бы существенно упростила их труд.

Инженерами QA Team было создано решение, которое позволяет в полной мере организовать данный процесс. Решение взаимодействует с API следующих систем и связывает их между собой: Redmine, Proxmox Virtual Environment, Testlink и систем. Основные требования, которые были поставлены для данной системы, стали:

- автоматическое создание задач в системе управления проектами и задачами Redmine, представленных для тестирования сборочных заданий в стабильные репозитории;
- назначение наблюдателей в созданные задачи по умолчанию;
- генерация сводного отчёта в Redmine, в котором содержится необходимая информация о задании, результаты прогонов автоматического тестирования, данные о существующих зависимостях, результат пересборки зависимых пакетов с заданием, наличие файловых конфликтов;
- автоматизация пересборки зависимых пакетов;
- разворачивание виртуальных машин в Proxmox Virtual Environment для тестирования сборочного задания: проверка обновления и установки пакетов несколькими способами, загрузка и запуск автоматизированных тестов с целью выявления регресса;
- отправка информации об автоматических прогонах в Testlink и на email инженера QA, работающего над заданием;
- взаимодействие с актуальной информацией о пакетах, используя ALTRepo API.

Текущий доклад описывает реализацию данного решения.

Литература

[1] ALT Linux Wiki, `https://www.altlinux.org/`

[2] Документация Redmine API, `https://www.redmine.org/projects/redmine/wiki/rest_api`

[3] Документация Proxmox VE API, `https://pve.proxmox.com/pve-docs/api-viewer`

[4] Bash Reference Manual, `https://www.gnu.org/software/bash/manual/bash.html`

Михаил Черноног
Обнинск, ООО «Базальт СПО»
https://packages.altlinux.org

Автоматическое тестирование дистрибутивов ОС «Альт» с использованием OpenQA

Аннотация

В докладе будет рассмотрено решение по автоматизации тестирования установки дистрибутивов ОС Альт с использованием открытого тестового фреймворка OpenQA, реализованное в компании Базальт СПО

Установка дистрибутива является первым, с чем сталкивается пользователь при взаимодействии с системой. Именно поэтому нужно свести к минимуму количество ошибок на данном этапе, ведь очень важно не испортить первое впечатление от ОС.

Одной из главных проблем в тестировании установки дистрибутива является время QA-инженера, поскольку требуется проверить весь процесс от начала и до конца на определённом наборе параметров. К примеру, в зависимости от выбранных настроек, для одного языка может насчитываться более 60 вариантов установки, такие как автоматическая установка или установка в ручном режиме при создании RAID0 + LVM.

Автоматизированное тестирование — это метод тестирования программного обеспечения, который выполняется с использованием специальных программных средств. Фреймворк OpenQA позволяет запускать собранный образ в виртуальной машине, эмулирует нажатия клавиш и движения мыши и сравнивает полученные скриншоты загрузки и установки ОС, а также работы приложений с эталонными изображениями.

Цель автоматизации — уменьшить количество тестовых примеров, которые нужно запускать вручную, а не полностью исключить ручное тестирование. Таким образом, существенно сокращается общее время тестирования.

Инженеры QA Team компании «Базальт СПО» постоянно разрабатывают новые сценарии автоматизированных тестов, а также поддерживают уже существующие тесты в актуальном состоянии.

Новые тесты создаются для проверки изменений, связанных с недавно добавленным функционалом. Сопровождение и поддержка автотестов осуществляется с каждым последующим циклом выпуска.

Подробности процесса автоматизации тестирования дистрибутива мы рассмотрим в текущем докладе:

- Что такое автоматизированное тестирование
- Framework OpenQA для автоматизации GUI
- Этапы создания тестов
- Дальнейшее развитие инфраструктуры

Литература

[1] Про Тестинг, `https://protesting.ru/automation/`

[2] openQA Documentation, `https://open.qa/docs/`

[3] Автоматизированное тестирование: что это? `https://logrocon.ru/news/automation_testing`

Ставцев Роман Геннадьевич
Москва, ООО «Базальт СПО»

Проблемы совместимости про построении аппаратных платформ на примере SoC Baikal-M

Аннотация

Появление доступных процессоров с архитектурой, отличной от x86, открыло новые возможности для разработчиков аппаратных решений.

В связи с этим появляется интерес к запуску ОС «общего назначения» на вновь создаваемых аппаратных решениях. Рассмотрим проблемы, которые могут возникать с совместимостью у разных аппаратных платформ при использовании процессора одного типа. Какие подходы возможны к решению проблем, которые возникают при запуске ОС.

Ставцев Роман Геннадьевич
Москва, ООО «Базальт СПО»

Перенос ROS с Ubuntu на решения «Базальт СПО»

Аннотация

Robotic Operating System (ROS) это активно развивающийся платформа (фреймворк) для разработки программного обеспечения роботов. Состоит из набора разнообразных инструментов, библиотек и определённых правил, которые упрощают решение задач разработки ПО управления роботами. ROS представляет собой надстройку над ОС. Исторически сложилось, что основной ОС для ROS является Ubuntu.

Дополнительно доступны и другие ОС, Debian, Arch Linux, появляется поддержка Windows и macOS. Основным методом распространения являются бинарные пакеты, также возможна установка из исходных кодов. Рассмотрим возможность сборки ROS1 из исходных кодов в среде ОС Альт.

Евгений Синельников
Саратов, ООО «Базальт СПО»

Проект: Доменная инфраструктура
`https://www.altlinux.org/Domain_Infrastructure`

Сценарии миграции доменных инфраструктур

Аннотация

Подходы к миграции службы каталогов и сопутствующих инфраструктурных служб предприятий на свободные решения требуют тщательного анализа. Исходной службой каталогов, для которой миграция актуальна, в большинстве случаев является Microsoft Active Directory. Данный доклад посвящён разбору основных сценариев миграции таких доменных инфраструктур на свободные инфраструктурные решения на базе дистрибутивов «Альт» и Sisyphus.

Миграция на свободные решения в корпоративной сетевой инфраструктуры, в первую очередь — это замена операционных систем клиентов и серверов под управлением операционных систем семейства ОС Windows» базе инфраструктуры Microsoft Active Directory (MS

AD) и на аналогичные по функциональным возможностям свободные операционные системы и инфраструктурные решения.

На текущий момент сложилось два фундаментально разных подхода к миграции доменной инфраструктуры на базе MS AD — «плавный» перевод инфраструктуры на Samba Active Directory или постепенная «миграция» на любое, аналогичное по функциональным возможностям инфраструктурное решение. Среди основных из таких решений стоит выделить два:

- Проект Samba, как полноценная замена MS AD на уровне протоколов и поддержки Windows-клиентов;
- Проект FreeIPA, как альтернативная реализация доменной инфраструктуры для Linux-клиентов.

«Плавная» миграция

Сценарий перевода инфраструктуры на базе MS AD даёт возможность сохранить уже сложившуюся, отлаженную годами доменную инфраструктуру не меняя конфигурацию множества рабочих станций (автоматизированных рабочих мест), а также серверов под управлением операционных систем на базе Windows, не пересоздавая множества пользователей, которые могут продолжать работу в том же окружении. В том числе и на новых рабочих станциях под управлением отечественных операционных систем. Миграция серверов, в некоторых случаях, также проводится постепенно. Такую возможность может обеспечить только совместимый с MS AD проект Samba. Данный сценарий предполагает следующий алгоритм действий:

- вывод из эксплуатации и замену сопутствующих инфраструктурных служб, работающих только с MS AD (MS Exchange, Sharepoint и д.р.);
- перенос сопутствующих системных служб, не относящихся непосредственно к доменной инфраструктуре (dhcp-серверов, центров сертификации и т.п.), с Windows-серверов на Linux сервера;
- временный, прозрачный для клиентов, перевод инфраструктуры в гибридное состояние через установку контроллеров домена на базе Samba в текущую инфраструктуру (в текущий домен);
- миграцию всех системных (FSMO и иных) ролей с контроллеров домена на базе MS AD на контроллеры домена на базе Samba;

- штатное отключение всех контроллеров на базе MS AD.

У данного сценария имеется огромное количество ограничений и множество рисков, которые сложно минимизировать, не затрагивая уже развёрнутые службы. Этот сценарий подходит для простых и консервативных конфигураций. Ключевые, документированные ограничения:

- схема леса 2008R2;
- репликация в домене должна выполняться против контроллера на базе Windows Server 2008R2 с тремя FSMO-ролями (хозяин схемы, инфраструктуры и PDC-эмулятор);
- объекты с парными атрибутами из расширенной схемы могут вызвать проблемы репликации;
- процесс первичной репликации для крупных баз (как правило, с большими по объёму бинарными атрибутами) занимает значительное время (до нескольких десятков часов).

Устранение этих ограничений требует специальной разработки, которые требуют значительных вложений и времени на отладку. Поскольку регулярных тестов для конкретных конфигураций не проводится, как правило, такие конфигурации находятся в закрытых сетевых контурах и недоступны для разработчиков, точный набор параметров для «плавной» миграции в проекте Samba на текущий момент оперативно не отслеживается.

«Постепенная» миграция

При отсутствии возможности перенести базу компьютеров, пользователей, групп и других объектов AD полностью, как есть, на другой «носитель», миграция доменной инфраструктуры проводится «постепенно». Такая миграция предполагает несколько промежуточных стадий в зависимости от целей:

- **миграция служб** (в первую очередь веб-приложений) в отдельную доменную инфраструктуру с доверительными отношениями с текущей;
- **миграция клиентских рабочих** мест на отечественные операционные системы;
- **комплексная миграция** серверов и клиентских рабочих станций.

При этом создаётся параллельная доменная инфраструктура, исходная пользовательская база (пользователи и группы) сохраняется через доверительные отношения и «постепенно» (по частям) переносится в новую инфраструктуру.

В случае миграции служб перенос пользовательской базы откладывается, сохраняются исходные рабочие места под управлением Windows, а все усилия прилагаются к тому, чтобы вывести из эксплуатации и заменить сопутствующие системные службы, не относящиеся непосредственно к доменной инфраструктуре.

В случае миграции рабочих мест главный упор делается на решение задачи замены пользовательских приложений, на их возможность запуска и работоспособность в новом окружении (прежде всего специализированного ПО, веб-приложений с «аутентификацией в домене», а также доступность почты, календаря и т. п.).

Комплексная миграция предполагает комбинирование миграции служб и рабочих мест и является наиболее трудоёмкой единовременной процедурой. Конкретные целевые особенности такой миграции зависят от специфики переносимой инфраструктуры.

Реализация сценариев миграции в дистрибутивах «Альт»

В дистрибутивах семейства «Альт» клиентские и серверные настройки разделены. Основным инструментом управления является «Альтератор», как центр управления системой. На текущий момент основной упор поддержки рассмотренных сценариев миграции сделан на интеграцию клиентских рабочих мест в инфраструктуру Active Directory. Интеграция реализуется в рамках сведения различных компонент управления операционной системой в целостный системный интерфейс, включающий в себя:

- обобщённый инструментарий подключения клиентов доменной инфраструктуры к различным доменным решениям (system-auth);
- групповые политики, как встроенный инструментарий управления конфигурациями [1];
- комплект графических средств администрирования, позволяющий заменить родной инструментарий управления Active Directory — Remote Server Administration Tool (RSAT) [2].

Литература

[1] ALT Linux Team, Групповые политики в решениях ALT, 2019, https://www.altlinux.org/Групповые_политики

[2] ALT Linux Team, Active Directory Management Center, 2022, https://www.altlinux.org/ADMC

Синельников Валерий
Саратов, ООО «Базальт СПО»

Проект: Текущий статус и перспектива развития Групповых политик в ОС«Альт» https://github.com/altlinux/gpupdate

Сценарии миграции доменных инфраструктур

Аннотация

Групповая политика — это набор правил, в соответствии с которыми производится настройка рабочей среды относительно локальных политик по умолчанию. Групповые политики работают в рамках домена, где их создают системные администраторы. Групповые политики в реализации MS Active Directory — это часть интегрированного решения, в составе которого политики хранятся в каталоге Sysvol на контроллерах домена. Проект Samba является альтернативным решением Active Directory под Linux/Unix. В целом проект Samba не поддерживает работу с групповыми политиками в конкретных дистрибутивных решениях. В данном докладе рассмотрим интеграцию групповых политик в операционной системе (ОС) «Альт» на основе проекта Samba.

Групповые политики в ОС «Альт» — это комплексное решение, которое включает хранение политик и шаблонов в каталоге Sysvol на контроллере домена, инструменты управления политиками, и механизмы применения настроек для компьютеров и пользователей.Централизованное управление и настройка парка машин с ОС «Альт» производится через инструмент `gpupdate`. Этот инструмент применяет групповые политики на рабочих станциях как на системном уровне, так и для отдельных пользователей. `gpupdate` поддерживает работу в доменной инфраструктуре MS AD или Samba AD. Цель доклада — обзор механизмов управления настройками систем, их функциональных возможностей в рамках проекта `gpupdate` для

ОС «Альт». Доклад описывает особенности `gpupdate`, а также графические инструменты для работы с каталогами Active Directory и Sysvol — ADMC и GPUI.

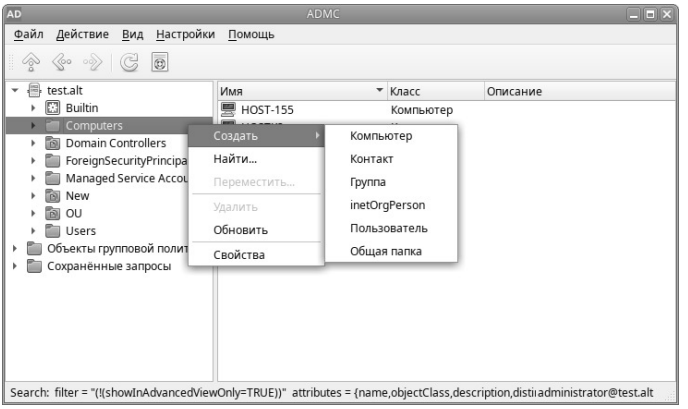


Рис. 1: ADMC — создание объекта.

Доменная инфраструктура

Для демонстрации групповых политик на рабочей станции или сервере, необходимо включить их в домен. Ввод компьютера в развёрнутой доменной инфраструктуре осуществляется согласно рекомендациям (описание на «ALT Linux Wiki»[1]). Компьютер следует предварительно настроить — установить в качестве домен-сервера ip-адрес домен-контролера.

На данный момент реализованы или будут реализованы в ближайшем времени применения следующих настроек:

- Стабильные:
  - Включение или выключение различных служб (сервисов systemd)
  - Управление control framework
  - Управление Gsettings

[1] `https://www.altlinux.org/Групповые_политики`

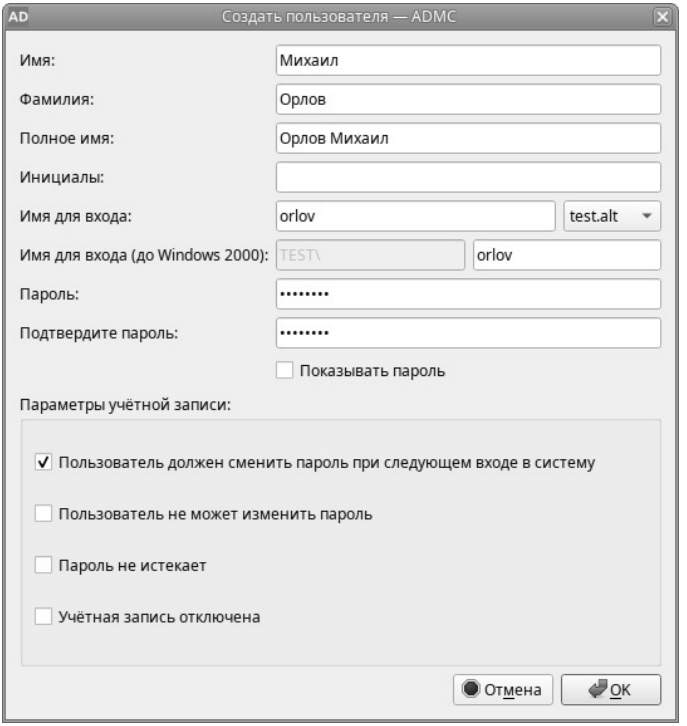


Рис. 2: ADMC – создание пользователя.

  - Генерация ярлычков запуска программ
  - Создание директорий
  - Запрет на подключение внешних носителей данных
- Экспериментальные:
  - Управление настроек браузера Firefox
  - Управление настроек браузера Chromium
  - Подключение сетевых дисков
  - Установка программного обеспечения
- Разрабатываемые:
  - Запуск скриптов при (запуске/завершение)

– Запуск скриптов при (вход/выход из системы)
– Управление ini файлами
– Управление общими каталогами
– Управление файлами (создание/удаление/пересоздание)

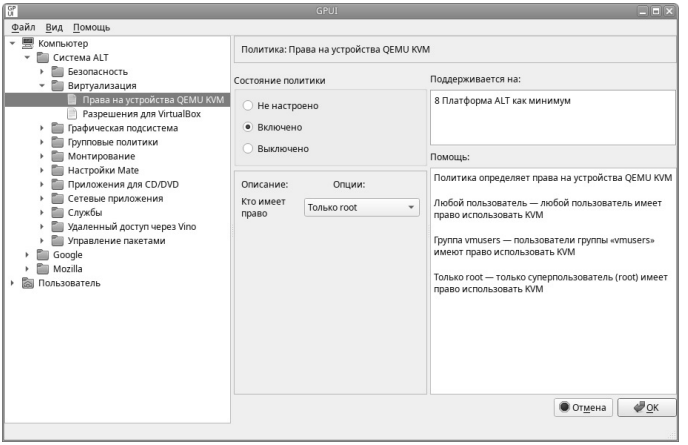


Рис. 3: GPUI — редактирование политики доступа к KVM.

Вспомогательные утилиты для администрирования

Администратору для формирования настроек необходимо использовать специальные инструменты. Инструмент, который в структурированном виде отобразит текущее состояние объектов домена. И инструмент для гибкой работы с групповыми политиками в домене.

В ОС «Альт» разработаны соответствующие графические приложения, которые позволяют решать выше озвученные задачи, аналогично оснастке RSAT в Windows.

Для управления базой данных конфигурации используется инструмент ADMC[2].

Для редактирования настроек клиентской конфигурации используется инструмент GPUI[3]:

[2] `https://www.altlinux.org/Групповые_политики/ADMC`
[3] `https://www.altlinux.org/Групповые_политики/GPUI`

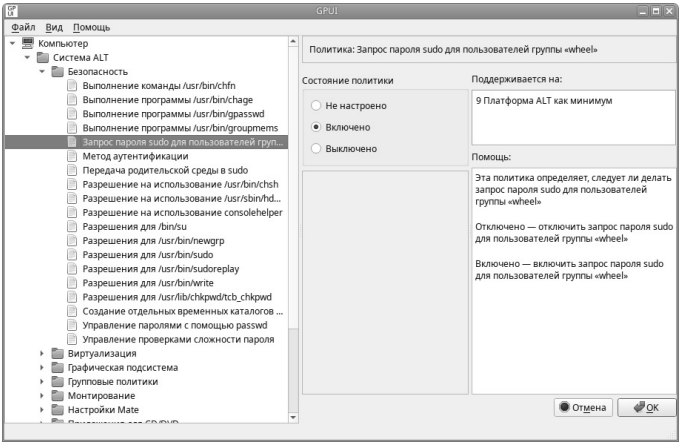


Рис. 4: GPUI — Редактирование политики sudo.

Вывод

Инженеры компании «Базальт СПО» разработали инструменты управления и механизмы обеспечения групповых политик в домене Active Directory. Инструмент `gpupdate` применяет групповые политики на рабочих станциях и серверах с установленной ОС «Альт». Графическое приложение ADMC управляет базой данных конфигурации Active Directory. Графическое приложение GPUI редактирует конфигурацию объектов групповых политик в домене.

Литература

[1] «Групповые политики» `https://www.altlinux.org/Групповые_политики`

[2] «ADMC» `https://www.altlinux.org/Групповые_политики/ADMC`

[3] «GPUI» `https://www.altlinux.org/Групповые_политики/GPUI`

Иван Савин
Саратов, ООО «Базальт СПО»

Проект: alterator-roles, libnss-role
http://git.altlinux.org/gears/a/alterator-roles.git,
https://github.com/Etersoft/libnss-role

Группы в группах

Аннотация

Данный доклад посвящён механизму, обеспечивающему назначение дополнительных групп для пользователей с помощью специализированного модуля. С точки зрения пользователя, этот механизм действует аналогично добавлению группы в группу подобно тому, как это реализовано в MS Windows. Рассмотрены особенности реализации и реальный опыт применения на практике.

Специализированный модуль — модуль ролей (libnss-role) — это модуль для службы переключения имён (NSS). Он делает возможным добавление групп в группы. Для администрирования модуля ролей существуют специальные вспомогательные утилиты, в рамках которых все группы условно разделены на две категории: роли и привилегии.

Роли — группы, предназначение которых указывать на характерную деятельность, выполняемую пользователем. Такими группами могут быть группы `localadmins`, `users`, `powerusers`, `developers` и др.

Привилегии — обычные группы, которые могут быть назначены пользователю. После этого пользователь может совершать действия, соответствующие данной привилегии. К привилегиям относятся такие группы как `cdwriter`, `audio`, `serial`, `virtualbox` и др.

В конфигурационном файле `nsswitch.conf` в цепочке применяемых модулей `role` должен быть последним:
`group: files systemd role`

Модуль выполняет обход всех групп, назначенных пользователю до применения этого модуля, и вычисляет дополнительные группы, которые должны быть назначены на основании правил, прописанных в файлах конфигурации.

Файлы конфигурации

Для хранения и получения информации о ролях и привилегиях модуль использует файл /etc/role. Начиная с версии 0.5.0, также поддерживается каталог /etc/role.d, позволяющий устанавливать отдельные конфигурационные файлы для ролей. Файлы конфигурации хранят информацию о вхождении групп в группы. Каждая строка в файлах имеет формат:

```
<имя_группы>:<имя_группы>[,<имя_группы>]*
```

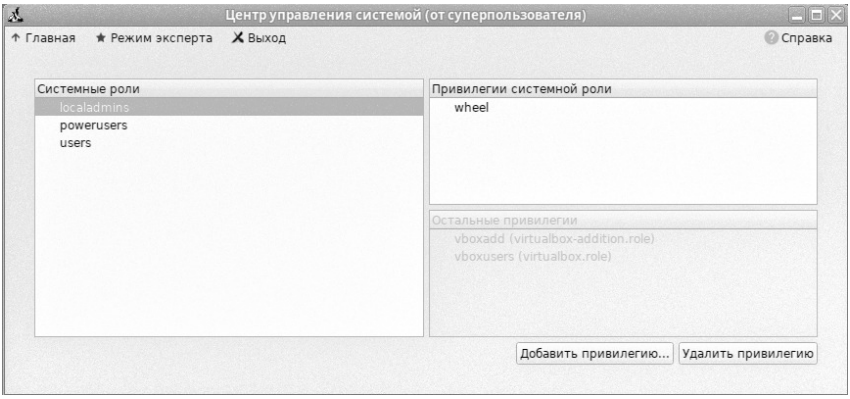


Рис. 1: Центр управления. Системные роли.

Имя до «:» означает, что данная группа будет являться ролью или что то же самое будет входить в другие группы, а последующие имена — в какие группы входит данная или какие привилегии ей назначены. Вхождения групп в группы применяется рекурсивно.

Пример:

Пусть у нас есть пользователь user. Пусть в файле /etc/group имеются записи:

```
group1:x:1:user
group2:x:2:user
group3:x:3:
group4:x:4:
```

```
group5:x:5:
group6:x:6:
```

А файл `/etc/role` содержит:

```
group2:group3,group4
group4:group5,group6
```

Тогда пользователь user получит все имеющиеся группы. Группы `group1` и `group2`, так как они назначены ему непосредственно, `group3` и `group4` — потому что они назначены группе `group2`, а группы `group5` и `group6` — так как они назначены группе `group4`.

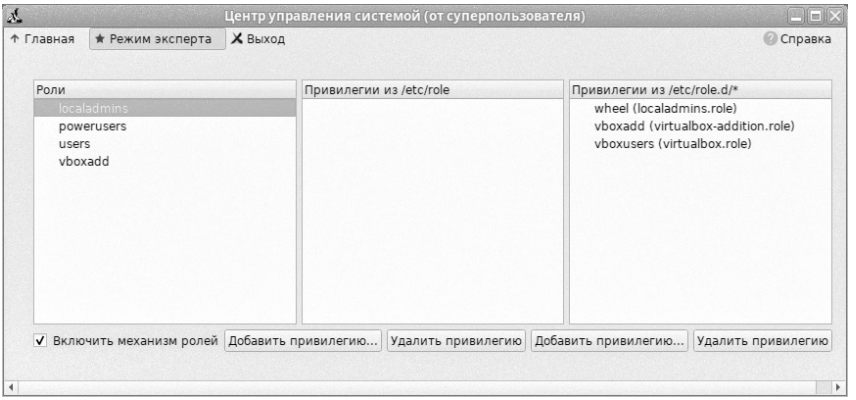


Рис. 2: Центр управления. Роли.

Вспомогательные утилиты для администрирования

roleadd — добавляет роль (если её ещё нет) и назначает ей привилегии. Пользователи, входящие в группу-роль, входят во все группы-привилегии.

roledel — удаляет привилегии из ролей, т.е. пользователи из группы-роли исключаются из удаляемой группы-привилегии.

rolelst — показывает текущий список ролей и привилегий.

На основании вышеприведённых утилит сделан графический инструмент, модуль альтератора *alterator-roles*. С его помощью можно манипулировать ролями и привилегиями более наглядно.

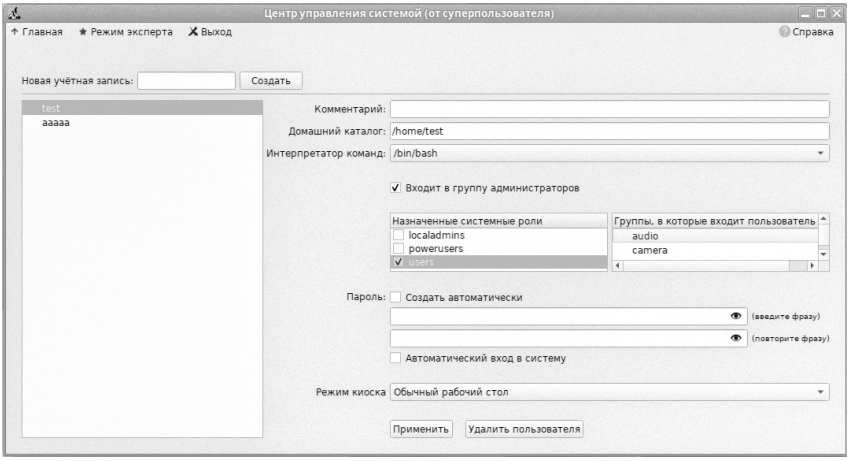


Рис. 3: Центр управления. Локальные учётные записи.

Системная роль (заголовок на первом рисунке) — это понятие, введённое для удобства. Такая роль имеет отдельный конфигурационный файл в /etc/role.d с названием соответствующей роли. Например, для системной роли *users* будет конфигурационный файл /etc/role.d/users.role. Такие роли можно будет распространять, например, групповыми политиками, и их было решено выделить в отдельное подмножество.

Одним из ключевых «удобных» моментов является возможность задать системные роли локальным пользователям из предопределённого списка. Системные роли интегрированы в управление локальными учётными записями (локальными пользователями).

Литература

[1] Модуль ролей `https://www.altlinux.org/Модуль_ролей`

[2] Alterator-roles `https://www.altlinux.org/Alterator-roles`

Вера Благовещенская
Обнинск, ООО «Базальт СПО»
https://github.com/altlinux/gpupdate

Тестирование инструментов управления групповыми политиками Active Directory и инструментов администрирования, совместимых с Microsoft Active Directory в ОС «Альт» в рамках гранта РФРИТ

Аннотация

Доклад посвящён обзору процесса тестирования приложений в ОС «Альт», на развитие которых выдан грант РФРИТ. К этим приложениям относятся модуль панели управления операционной системы для включения механизма применения конфигурации на клиентских машинах (alterator-gpupdate), модуль клиентской машины для применения конфигурации (gpupdate), модуль удалённого управления базой данных конфигурации (admc), модуль редактирования настроек клиентской конфигурации (gpui).

В декабре 2021 года компания «Базальт СПО» оказалась в числе победителей конкурсного отбора Российского фонда развития информационных технологий (РФРИТ) среди проектов по разработке российских решений в сфере ИТ.

Грант выдан на развитие инструментов управления групповыми политиками, которые входят в состав операционных систем «Альт Сервер», «Альт Образование», «Альт Рабочая станция».

Система управления групповыми политиками производства «Базальт СПО» расширяет функционал штатных политик Microsoft^{TM} Active Directory собственными шаблонами операционных систем «Альт». Эти инструменты входят в топ-лист требований корпоративных заказчиков к функционалу российских операционных систем.

Используя систему управления групповыми политиками производства «Базальт СПО», организации могут интегрировать компьютеры с ОС «Альт» в унаследованную инфраструктуру под управлением Windows и управлять компьютерами с обеими ОС одновременно и единообразно.

В 2022 году разработчики «Базальт СПО» усовершенствуют инструментарий управления групповыми политиками, обеспечив заказчикам возможность переводить на ОС «Альт» свою цифровую инфраструктуру.

Важным этапом жизненного цикла разработки любого программного обеспечения является тестирование. Чтобы получить высококачественное ПО и минимизировать затраты на исправление ошибок, необходимо как можно раньше начинать его тестировать, уже на стадии анализа требований.

Инженеры QA Team компании «Базальт СПО» приступили к тестированию на ранних стадиях разработки. Подробности этого процесса мы рассмотрим в текущем докладе:

- Как создавались тестовые сценарии.
- Какие виды и типы тестирования были выполнены на первом этапе работ по гранту.
- Статистика найденных и исправленных ошибок.
- Привлечение к тестированию инженеров с различным опытом работы.
- Взаимодействие с разработчиками.

Литература

[1] Джоэл Х. Спольски, «Джоэл о программировании»

[2] Про Тестинг, https://protesting.ru/testing/

[3] Групповые политики https://www.altlinux.org/Групповые_политики, https://www.altlinux.org/Групповые_политики/GPUI, https://www.altlinux.org/Групповые_политики/ADMC.

Сергей Мартишин, Марина Храпченко
Москва, Институт системного программирования им. В.П. Иванникова РАН
https://github.com/otd13isp/Pereslavl2022/tree/main

Визуализация агрегированных конфиденциальных данных в научных проектах в сети Интернет с использованием языка R

Аннотация

В научных проектах часто возникает проблема обработки и визуализации конфиденциальных данных. Симметрические запросы, позволяющие получить статистические данные являются дедуктивно безопасными. Для визуализации статистических данных в сети Интернет одним из наиболее удобных является язык R, имеющий большое количество встроенных функций для анализа и визуализации данных. Использование графических пользовательских интерфейсов и простота установки делают доступной работу с ним широкому кругу специалистов.

Хранение и обработка больших объёмов данных требует значительных аппаратных и программных ресурсов. Удобство использования и широкий спектр IT-услуг, предоставляемых облачными сервисами, которые обеспечивают возможность доступа с любого устройства и организации совместной работы, а также сэкономить на дорогостоящем оборудовании, привело к тому, что всё большее количество клиентов выбирает облачные сервисы для хранения и обработки информации.

Говоря о хранении и обработке информации на облаке, следует иметь в виду, что достаточно часто на облаке хранятся конфиденциальные данные, используемые совместно, работа с которыми осуществляется в соответствии с правами доступа. Это могут быть медицинские данные, данные об итогах голосования и т.п. Такого рода данные содержат значительный объём информации, анализ которой представляет интерес для научных сообществ, профильных специалистов-практиков, органов власти.

Ранее авторы в [1] исследовали модель облачных вычислений над конфиденциальными данными. В [2] было сформулировано определение дедуктивной безопасности: клиент базы данных, получив ответы на все доступные ему запросы, не может получить дедуктивно (в виде логического следствия) конфиденциальные данные пользователей. Соответственно, контроль полномочий клиента должен включать проверку дедуктивной безопасности тех запросов, с которыми клиент обращается к базе данных. Также в [2] было доказано, что симметрические запросы являются дедуктивно безопасными, поскольку призваны собирать лишь статистические данные (например: сумму, медиану, среднеквадратическое отклонение и др.) о количественных данных, содержащихся в базе, но не раскрывают никакой специальной информации (например, имён), относящейся к отдельным пользова-

телям базы данных [3]. Таким образом, вопрос сохранения конфиденциальности данных ранее был достаточно хорошо изучен.

Однако остаётся не менее важный вопрос представления полученных агрегированных данных в виде, удобном для восприятия человеком. Одним из наиболее широко используемых языков программирования для анализа и визуализации статистических данных остаётся интерпретируемый объектно-ориентированный язык R [4]. Он является СПО (свободное программное обеспечение) с открытым исходным кодом в рамках проекта GNU.

С помощью языка R можно обработать данные для исследования, работать с данными разных форматов, объединяя их из различных источников, визуализировать данные. Установка R под Linux выполняется стандартно:

dnf install R

Большим удобством является возможность использовать не только командную строку, но и графические пользовательские интерфейсы, которые также являются СПО и распространяются под свободной лицензией GNU GPL, например, среда разработки RStudio [5]. Для установки следует выполнить команды:

dnf install rstudio-desktop

dnf install rstudio-server

Основные статистические методы реализованы в качестве стандартных функций. Также для R разработано большое количество дополнительных пакетов, которые также являются СПО. Большая часть из них собраны в специальном репозитории CRAN (Comprehensive R Archive Network) [6], последние версии разрабатываемых пакетов можно также найти на GitHub. Из недостатков заметим, что не все пакеты хорошо задокументированы.

Установка пакетов выполняется при помощи функции `install.packages()`, с указанием имени пакета.

Например, команда `install.packages("ggplot2")` устанавливает пакет для построения графики, а `install.packages("plumber")` устанавливает пакет, позволяющий преобразовать существующий код R в веб-API, добавляя специальные комментарии [7].

Также большим удобством является возможность получить сведения об используемой функции различными способами: в RStudio можно поставить курсор на название функции и нажать F1, также

можно перед названием функции напечатать один или два знака вопроса и выполнить запрос.

Для запуска проекта необходимо установить сервер. Для простоты будем использовать сервер Apache. Его установка и запуск выполняются стандартно:

dnf install httpd

systemctl start httpd.service

Для подсоединения к серверу необходима программа для управления фаерволом необходимо становить пакет system-config-firewall (# dnf install system-config-firewall-base.noarch), вызвать программу (# firewall-config) и поставить галочку напротив http, выбрав конфигурацию «Рабочие» или «Постоянная».

В файле программы на языке R следует обратить внимание на строку, которую необходимо описать для доступа к программе (#* @post /точка_входа_на_сервер). Она сообщает о том, что запрос будет осуществляться, например к `http://localhost:8000/точка_входа_на_сервер` методом POST.

Для задания размера возвращаемого изображения используется строка:

#* @serializer png list(width = 800, height = 400)

Таким образом, использование языка R для визуализации в сети Интернет статистических данных в научных проектах, связанных с обработкой конфиденциальных данных, представляется предпочтительным, поскольку имеет большое количество встроенных функций для анализа и визуализации данных, его установка является стандартной, а работа с ним значительно упрощается при использовании R Studio. Базовый код проекта представлен в `https://github.com/otd13isp/Pereslavl2022/tree/main`.

Литература

[1] Варновский Н. П., Мартишин С. А., Храпченко М. В., Шокуров А. В. // Пороговые системы гомоморфного шифрования и защита информации в облачных вычислениях, Программирование. 2015. № 4. с. 47–51.

[2] Варновский Н. П., Захаров В. А., Шокуров А. В., О дедуктивной безопасности запросов к базам конфиденциальных данных в системе облачных вычислений // Вестник Московского университета. Серия 15: Вычислительная математика и кибернетика. 2017. № 1. с. 38а–44.

[3] Мартишин С. А., Храпченко М. В. Организация облачных вычислений над конфиденциальными данными на СПО. // Материалы конференции / АНО «Национальный суперкомпьютерный форум», Институт Программных Систем РАН, Институт Логики, Базальт СПО. — Переславль, 7–9 февраля 2020 года / отв. Ред. Чёрный В.Л. — М.: МАКС Пресс, 2020. с. 171–174

[4] The R Project for Statistical Computing [Электронный ресурс] — Режим доступа: `https://www.r-project.org`, свободный. Яз. англ. — Дата обращения: 11.03.2022.

[5] RStusio [Электронный ресурс] — Режим доступа: https://www.rstudio.com, свободный. Яз. англ. — Дата обращения: 11.03.2022.

[6] Fedora Packages of R Software [Электронный ресурс] — Режим доступа: `https://cran.r-project.org/bin/linux/fedora`, свободный. Яз. англ. — Дата обращения: 11.03.2022.

[7] Plumber [Электронный ресурс] — Режим доступа: `https://www.rplumber.io`, свободный. Яз. англ. — Дата обращения: 11.03.2022.

Иван Викторович Шишунов
Талдом, МОУ Квашёнковская СОШ

Организация занятий по программированию на базе ОС «Альт Образование»

Аннотация

Рассказ о внедрении в МОУ Квашёнковской СОШ модели внеурочной деятельности по программированию на всех трёх ступенях образования на базе российской операционной системы «Альт образование» с возможностью как очного, так и дистанционного обучения. Эта модель может быть взята за основу и масштабирована в других общеобразовательных организациях.

Конечно, даже в нашу цифровую эпоху не каждый человек станет it-специалистом или системным администратором, но, хотим мы того или нет, всем нам в своей жизни, так или иначе, приходится заниматься программированием. Например, прежде, чем выразить свою мысль, её необходимо разложить на простые смысловые единицы и выстроить в определённой последовательности, то есть, формулируя

фразу, мы, по сути, пишем для себя программу, которую сами же потом выполняем. «Мы живём в мире программ, и сами постоянно программируем, не осознавая этого» — эти слова советского академика А.П.Ершова из его известной статьи «Программирование — вторая грамотность»[1] как нельзя лучше подходят для описания повседневной деятельности человека и окружающего его мира. И всё же никогда ещё за более чем 30 лет своего существования эти слова не звучали так сильно, как сегодня, на третьем десятке 21-ого века, когда человек имеет возможность общаться с голосовыми помощниками и чатботами, ездить на беспилотном транспорте и жить в «умном» доме. Казалось бы, такому важному и актуальному навыку, как умение программировать, в школах должно уделяться много внимания, но реальность такова, что в обычной общеобразовательной школе программирование изучается только в рамках базового курса информатики, на который отводится всего 1 час в неделю, причём начиная с 7-ого класса. Мы не будем сейчас выяснять и обсуждать причины, почему дела обстоят именно так. Гораздо важнее разобраться, как выйти из сложившегося положения и предоставить возможность школьникам на всех ступенях обучения заниматься программированием, обеспечив при этом не только формирование базовых навыков алгоритмизации, но и возможность углублённого изучения материала и применения полученных знаний при решении реальных практических задач. При этом также следует учесть некоторые обстоятельства, в которых в последнее время оказались школы. Во-первых, в современных эпидемиологических условиях учитель должен быть готов проводить занятия как в очном, так и в дистанционном формате. Во-вторых, на школы распространяется Постановление Правительства РФ[2] о запрете закупок иностранного программного обеспечения, в том числе операционной системы Windows, в связи с чем новые компьютеры в образовательные организации поставляются с предустановленными российскими операционными системами семейства Linux.

Наша школа для решения вышеозвученной задачи пошла по пути организации кружков внеурочной деятельности. Летом 2021 года я успешно прошёл конкурсный отбор учителей, проводимый Благотворительным Фондом развития образования «Айкью Опшн»[3], и стал участником проектов «Мир Scratch» и «Поколение Python», что позволило мне в начале этого учебного году организовать два кружка по программированию: на языке Scratch для 3-5 классов и на языке Python для 6-9 классов. Ещё один кружок — для учащихся 10-11

классов — появился в начале 2021, когда наша школа стала площадкой программы «Intel AI For Youth: Технологии ИИ для каждого».На занятиях этого кружка старшеклассники не просто изучают язык программирования Python, но и учатся его применять при выполнении различных практических кейсов по таким направлениям, как обработка естественного языка, анализ данных и машинное зрение.

В рамках договора о сотрудничестве с компанией «Базальт-СПО» на всех ноутбуках центра «Точка роста», где проходят занятия по программированию, установлены российские операционные системы «Альт Образование». Я имею большой опыт использования операционных систем семейства Linux и другого свободного программного обеспечения в решении различных образовательных задач, поэтому для меня не составило труда организовать учебный процесс в кружках по программированию в этих условиях. На занятиях в кружке «Мир Scratch» ребята используют программу Scratch Desktop, при этом также имеется возможность с помощью браузеров Chromium или Mozilla Firefox работать в режиме онлайн на сайте scratch.mit.edu[4]. При изучении языка Python на локальных компьютерах используются системы программирования IDLE и Wing, но большую часть работы учащиеся делают прямо в браузерах на платформе Stepik[5]. На занятиях по программе «Intel AI For Youth: Технологии ИИ для каждого» обычно используется онлайн-сервис Google Collaborate[6], позволяющий работать с так называемыми Jupiter-ноутбуками — файлами, включающие в себя фрагменты кода на Python, которые можно запускать в произвольном порядке и сразу видеть результаты. Но работать с Jupiter-ноутбуками можно также и на локальных компьютерах без выхода в интернет. Для этого в Альт Образовании нужно установить несколько дополнительных пакетов из репозитория.

Согласно стратегии научно-технологического развития РФ[7], утверждённой Указом Президента в декабре 2016 года, целью научно-технологического развития РФ является обеспечение независимости и конкурентоспособности страны, а также лидерство по избранным направлениям в рамках традиционных и новых рынков технологий, продуктов и услуг. Программирование является неотъемлемой частью практически всех актуальных научно-технологических направлений, поэтому от того, как оно будет изучаться в школе, зависит ни много ни мало будущее научно-технологического развития нашей страны. Но стоит отметить, что для решения любой задачи нужны исходные данные. Если мы говорим про обычную общеобразовательную школу,

то для повышения качества образования по дисциплинам, связанным с программированием, необходимы как минимум: заинтересованный учитель, нацеленный на постоянное саморазвитие, школьная администрация, готовая оказывать такому учителю всяческую поддержку, а также необходимое материально-техническое и программное обеспечение, позволяющее сделать образовательный процесс гибким и комфортным. Любая школа, имеющая в своём распоряжении такой набор исходных данных, может внедрить модель, которая реализуется у нас: внеурочная деятельность по программированию на всех трёх ступенях образования на базе российской операционной системы с возможностью как очного, так и дистанционного обучения.

Литература

[1] А.П.Ершов. Статья «Программирование — вторая грамотность». http://ershov.iis.nsk.su/ru/second_literacy/article

[2] Постановление Правительства Российской Федерации от 16 ноября 2015 года № 1236 «Об установлении запрета на допуск иностранного программного обеспечения при закупках для государственных и муниципальных нужд». https://base.garant.ru/71252170/

[3] Благотворительные проекты фонда «Айкью Опшн» https://iqcharity.ru/projects

[4] Сайт среды программирования Scratch https://scratch.mit.edu/about

[5] Курс «Поколение Python: курс для начинающих» на платформе Stepik https://stepik.org/course/58852/info

[6] Сервис Google Colab https://colab.research.google.com/?hl=ru_RU

[7] Указ Президента РФ от 1 декабря 2016 г. N 642 «О Стратегии научно-технологического развития Российской Федерации» https://base.garant.ru/71551998/