ООО «Базальт СПО»
Институт Программных Систем РАН

Объединённая конференция «СПО: от обучения до разработки»

Переславль-Залесский, 15–18 июня 2021 года

Сборник тезисов конференции

Москва,
МАКС Пресс,
2021

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, 15–18 июня 2021 г. / отв. ред. Чёрный В. Л. — М. : МАКС Пресс, 2021. — 216 с.
ISBN 978-5-317-06629-1

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 № 029

ISBN 978-5-317-06629-1

Программа конференции

15 июня, вторник

Секция: СПО в учебном процессе

11.30-14.00 Заселение, обед, регистрация участников

13:30 Автобус от гостиницы Переславль

Дневное заседание
14.00–18.30

14.00 Приветственное слово организаторов

14.30–14.50 Н. Н. Непейвода

Возрождение университетского образования в Переславле . 11

14.50–15.10 Р. Л. Лашин

О переходе системы образования на отечественные решения 13

15.10–15.25 А. С. Кремер

Информационная культура цифровой трансформации 15

15.25–15.40 В. А. Кирсанов, Т. В. Черствов

Направления доработки ТОМИИТ 17

15.40–16.00 Е. С. Орлов

Программа профессиональной переподготовки ООО «Базальт СПО» и СПбПУ для системных администраторов ОС Альт 19

16.00–16.20 Д. Ю. Гудзенко

Специалист.Ру: кто учил Майкрософт, теперь учит Линукс 21

16.20–16.40 Кофе-пауза

19:30 Автобус в гостиницу Переславль

16 июня, среда

Секция: СПО в учебном процессе

09:30 Автобус от гостиницы Переславль

Утреннее заседание
10.00–13.00

Вечернее заседание
15.00–19.30

19:00 Автобус в Москву

17 июня, четверг

Секция: Разработка свободного ПО

09.30-10.00 Регистрация участников

09:30 Автобус от гостиницы Переславль

Утреннее заседание
10.00–13.00

11.20–11.40 Кофе-пауза

13:00 Автобус в гостиницу Переславль

13.00–15.00 Перерыв на обед

14:40 Автобус от гостиницы Переславль

Вечернее заседание
15.00–19.30

15.00–15.30 Н. А. Костригин

15.30–16.00 Э. Р. Хабирова, А. С. Анисимов

16.00–16.30 А. А. Савченко, И. А. Курдюков

16.30–16.50 Д. В. Дегтярев

16.50–17.10 Е. А. Синельников, И. И. Чудов

17.10–17.30 Кофе-пауза

17.30–17.50 И. Ю. Власенко

17.50–18.10 М. В. Быков

18.10–18.30 С. П. Левин

18.30–19.00 Круглый стол: «make-initrd: генератор initramfs и не только»

19:30 Автобус в гостиницу Переславль

18 июня, пятница

Секция: Разработка свободного ПО

09:30 Автобус от гостиницы Переславль

10.00–10.20 А. Г. Михеев

Свободная система управления бизнес-процессами и административными регламентами RunaWFE Free. Изменения, вошедшие в последние версии. 184

10.20–10.40 Р. А. Бородин

Почтовый (многофункциональный) сервер Tegu 189

10.40–11.00 Д. А. Винокуров

Дайджесты FOSS News и средства автоматизации их составления 194

11.00–11.20 А. А. Андреев

Распознавание речи на мобильных устройствах, управляемых Linux, на примере ОС Аврора 198

11.20–11.40 Кофе-пауза

11.40–12.00 В. А. Синельников

Служба alterator-dbus, как возможность представить API модулей центра управления системой ALT через D-Bus 199

12.00–12.20 И. В. Савин

Обновление ядра в ОС «Альт» 201

12.20–12.50 Л. А. Кривошеин

Инструменты начальной загрузки и массового развёртывания «Альт» 205

12.50–13.10 Б. А. Зоричев, Ю. Л. Попов

Разработка протокола связи для устойчивой передачи видео в самоорганизующихся сетях 208

13.10–13.30 Г. В. Шатров

Использование Open Source-подходов при построении Государственных ИС 211

14:00 Автобус в Москву

Вне программы

Секция: Разработка свободного ПО

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

Возрождение университетского образования в Переславле

В 1992 году возникло уникальное учебное заведение — Университет города Переславля (УГП). Его совместно организовали Институт программных систем АН СССР и администрация города. В университете был всего один факультет с двумя специализациями: информатика и экономика.

Особенностью университета была теснейшая связь образования с реальными проектами, при этом информатики работали по проектам ИПС РАН, находившимся на передовых рубежах науки. Это было осознанное решение по трём причинам.

Во-первых, информатика столь быстро развивается, что изучавшие новейшие системы на первом курсе выходят из вуза зная морально устаревшие.Как здесь поступить? Бежать образованию вместе с наукой и практикой. Быстро развивающийся процесс можно изучать лишь изнутри.

Во-вторых, ИПС РАН не мог прожить и делать одновременно амбициозные суперкомпьютерные, в областях медицинской информатики и ИИ системы и проекты, опираясь лишь на «иммигрантов», значит, надо было наладить подготовку кадров высокой квалификации в самом маленьком городе.

В-третьих, это позволяло максимально использовать в преподавании конкретные высококвалифицированные кадры института.

Опыт УГП показал, что реально решать кадровую проблему даже в небольшом городе. А для города университет давал престиж и ещё один эффект, который отметил министр образования Сирии и распространил опыт в своей стране. Если из маленького города уходят учиться в университет большого, то возвращаются приблизительно 0%. Если учатся на месте, то в городе остаются около 50%. В Переславле оставалось 57%, а в Ярославской области 2/3. Демографический кризис в городе замедлился.

Город вложился в университет льготной арендой трёх зданий в центре (правда, одно из них — общежитие — было изношено на 100%).

К 2011 году сложилась уникальная непрерывная система высококачественного и дешёвого образования. Детский сад «Почемучка», лицейский класс, летняя школа по информатике, УГП, аспирантура. И вся она была постепенно демонтирована к 2019 году. Особенно постарался предпоследний мэр Переславля Волков, который начал с продажи свалки в заповедном районе, затем отжал здания УГП, но так их и не использовал (сейчас есть проект в бывшем здании университете на первом этаже открыть ресторан, а на втором — номера. Я мрачно шутил, что вместо университета делают бордель) и наконец довёл невыплатами даже заработной платы городское хозяйство до полного развала. Дошло до того, что когда отключили от теплосети местных военных, пришли вежливые люди и сами включили себе тепло (почему-то это проигнорировали наши СМИ). Отток людей увеличился в два раза, в первую очередь за счёт молодёжи. В наказание его сделали мэром Ярославля.

Сейчас мы пытаемся возродить университетское образование в городе на базе Ярославского государственного университета имени П.Г. Демидова:

- Обучение проводится на базе очного направления 02.03.02 «Фундаментальная информатика и информационные технологии», профиль «Информатика и компьютерные науки».
- Обучение по специальным предметам (3, 4 курсы) проводится на базе Института программных систем имени А.К.Айламазяна РАН как ресурсного центра ЯрГУ

Направление **02.03.02 Фундаментальная информатика и информационные технологии, профиль Информатика и компьютерные науки**:

- Количество бюджетных мест — 45
- Проходной балл ЕГЭ на бюджетные места — 233
- Военная кафедра — нет
- Срок обучения (бакалавриат), лет — 4

Информация о направлении доступна на сайте ЯрГУ `https://www.uniyar.ac.ru/education/higher_education/6866/`

Конечно, это половинчатое решение, но будем надеяться, что это первый шаг к возрождению всей системы образования.

Ренат Лашин
Москва, АРПП «Отечественный софт»

О переходе системы образования на отечественные решения

Ассоциация Разработчиков Программных Продуктов «Отечественный софт» учреждена в 2009 году крупнейшими российскими разработчиками тиражируемого программного обеспечения. Членами Правления ассоциации являются: Наталья Касперская (председатель), Александр Голиков, Евгений Михалицын, Дмитрий Комиссаров, Алексей Смирнов, Андрей Ярных, Дмитрий Шушкин, Дмитрий Дырмовский и Андрей Голов (`https://www.arppsoft.ru/association/board-of-directors/`).

В настоящее время Ассоциация «Отечественный софт» насчитывает более 200 российских ИТ-компаний. Общий оборот членов АРПП в прошлом году в среднем составил 163 млрд. рублей, суммарная численность сотрудников — 36 750 человек. Среди актуальных направлений деятельности Ассоциации — импортозамещение, поддержка экспорта российского ПО, продвижение отечественных решений в образовании, содействие компаниям в привлечении финансирования. Формируются и другие точки роста. В начале 2021 года в Ассоциации были созданы 4 новых комитета: по цифровой трансформации, искусственному интеллекту, телекоммуникациям и информационной безопасности. Как отмечает Наталья Касперская, АРПП «Отечественный софт» — это единственная ассоциация, которая сегодня растёт. С каждым годом в её состав входит всё больше отечественных компаний, разрабатывающих программные продукты и решения. На площадке Ассоциации российские разработчики объединяются для совместной работы над ключевыми вопросами развития отрасли.

Благодаря инициативам АРПП «Отечественный софт», удалось сократить сроки перехода субъектов критической информационной инфраструктуры (КИИ) на российское ПО и оборудование на 1 год — на 2023 и 2024 годы соответственно. Также усилиями Ассоциации был сформирован перечень из 155 российских независимых решений, готовых к внедрению на объектах КИИ. Важным результатом работы также стало создание Каталога совместимости российского ПО, в ко-

тором сегодня насчитывается порядка 9 тысяч программных продуктов и 600 подтверждённых совместимостей.

Ассоциация «Отечественный софт» участвовала в разработке проекта так называемого второго пакета мер господдержки российской ИТ-индустрии, в который всего вошло 64 предложения по девяти тематическим блокам: общесистемные меры, отечественные решения для бизнеса, электронные образовательные сервисы, электронные медицинские сервисы, отечественное офисное ПО и ОС, обработка данных и облачные сервисы, решения в области искусственного интеллекта, больших данных и интернета вещей, производство отечественных компьютерных игр и российского профессионального видеоконтента, решения в сфере информационной безопасности. Среди включённых мер поддержки есть и такая, которая предусматривает переход школ на использование российского программного обеспечения (мессенджеры, почта, ВКС, офисное программное обеспечение, операционные системы). Однако в настоящее время сложности перехода образовательных учреждений на российское ПО возникают из-за следующих препятствий:

- невозможность организации и проведения ЕГЭ, ОГЭ и КЕГЭ без установки зарубежных ОС, поскольку это требование прописано в официальных руководствах пользователя, утверждённых и рассылаемых по пунктам проведения ЕГЭ, ОГЭ и КЕГЭ;
- невозможность работы с электронными формами учебников от ведущих издательств Просвещение-Союз, Академкнига и др., так как они совместимы только с зарубежными ОС.

Кроме того, нет никаких указаний на необходимость закупки российского ПО в рекомендациях по приобретению оборудования для обновления материально-технической базы общеобразовательных и профессиональных образовательных организаций в рамках поставок по национальному проекту «Образование» в рамках федерального проекта «Цифровая образовательная среда». Тем самым предлагается закупать оборудование с предустановленной ОС, без какого-либо указания на Реестр российского ПО или на отечественное происхождение. С просьбой временно приостановить не соответствующие требованиям импортозамещения закупки ПО и оборудования для образовательных учреждений в апреле текущего года обратились к вице-премьеру РФ Д.Н. Чернышенко Ассоциации «Отечественный софт» совместно с АРПЭ.

Также вторым пакетом мер поддержки ИТ-отрасли предусмотрено мероприятие, направленное на увеличение участия отечественных разработчиков в международных проектах развития свободного программного обеспечения. Как сообщил замглавы министерства Максим Паршин в рамках ПМЭФ-2021 Минцифры совместно с ИТ-компаниями к сентябрю разработают стратегию развития в России программного обеспечения с открытым кодом (open source). В документе предстоит определить, какие меры господдержки развития open source могут быть необходимы — в частности, уместны ли прямые финансовые меры, какое регулирование должно быть, потребуется ли поддержка через госзаказ. Министерство видит три основных приоритета стратегии: эффективность разработки, качество продуктов на основе open source разработок; технологическая и информационная безопасность, возможность проводить независимый аудит исходного кода. И третье — это технологическая независимость, это снижение санкционных рисков, снижение зависимости от глобальных вендоров...

Аркадий Кремер, Александр Иванюк, Елена Правидло
Москва, Общественно-государственное объединение «Ассоциация документальной электросвязи» (АДЭ)
http://rans.ru/

Информационная культура цифровой трансформации

Аннотация

В докладе обсуждается разработка комплекса отечественных аппаратно-программных средств и учебно-методической литературы для оснащения новой магистерской программы «Информационная культура цифровой трансформации», предлагаемой общественно-государственным объединением «Ассоциация документальной электросвязи». Целью программы является подготовка технических экспертов, способных представлять интересы Российской Федерации в международных организациях, работающих в сферах развития ИКТ-инфраструктуры и обеспечения доверия и безопасности при её использовании.

Целью магистерской программы нового профиля, разработанной общественно-государственным объединением «Ассоциация докумен-

тальной электросвязи» (АДЭ) по направлению 10.04.01 Информационная безопасность, является подготовка технических экспертов, способных представлять интересы Российской Федерации в международных организациях, работающих в сферах развития ИКТ-инфраструктуры и обеспечения доверия и безопасности при её использовании.

Открытие новой магистерской программы — это ответ на запрос государства и бизнеса о необходимости повышения эффективности участия российских экспертов в деятельности международных организаций по стандартизации. Участие в работе международных организаций:

- обеспечивает возможность изучения российскими экспертами лучших мировых практик,
- содействует развитию международного сотрудничества и продвижению российской продукции и отечественных технических решений на международные рынки, обеспечивая тем самым участие в международном разделении труда,
- обеспечивает имиджевые преимущества нашей стране.

В рамках двухгодичного обучения по новой магистерской программе осуществляется подготовка специалистов, способных применять полученные знания в сферах развития ИКТ-инфраструктуры и обеспечения доверия и безопасности при её использовании, разбирающихся в вопросах нормативного правового и технического регулирования, заинтересованных в разработке и использовании отечественных аппаратных и программных средств, разбирающихся в процедурных и организационных вопросах деятельности международных организаций по стандартизации, умеющих вести дискуссии на английском языке, имеющих навыки логического мышления и принятия решений, воспитанных этически и нравственно.

Приоритетным направлением подготовки является формирование способности применять полученные знания в интересах обеспечения информационной безопасности и технологической независимости Российской Федерации. Достижению этих целей призваны содействовать:

- учебно-исследовательский центр интернет-технологий для формирования компетенций в сфере развития ИКТ-инфраструктуры;

- учебно-исследовательский центр информационной безопасности для формирования компетенций в сфере защиты объектов критической информационной инфраструктуры;
- информационно-аналитическая система (ситуационный центр) кадрового обеспечения международного технологического сотрудничества;
- учебно-исследовательский центр по отработке навыков презентаций технических предложений и ведения дискуссий для аргументированного продвижения национальных интересов в условиях, максимально приближённых к работе экспертов в международных организациях.

Виктор Кирсанов, Вячеслав Малиночкин, Марс Магафуров, Тимофей Черствов
Москва, Московский Технический Университет Связи и Информатики (МТУСИ), Общественно-государственное объединение «Ассоциация документальной электросвязи» (АДЭ), кафедра «Технологии электронного обмена данными» (ТЭОД)
http://teod.rans.ru/

Направления доработки ТОМИИТ

Аннотация

В докладе рассматриваются перспективные направления доработки Типового отечественного модуля изучения интернет-технологий (ТОМИИТ): удалённый доступ, переход на новую версию операционной системы, возможные инфраструктурные вариации построения. ТОМИИТ с 2019 года введён в эксплуатацию на базовой кафедре Ассоциации документальной электросвязи в МТУСИ. Его отличительной особенностью является построение на базе отечественных аппаратно-программных средств.

Типовой отечественный модуль изучения интернет технологий (ТОМИИТ) с 2019 года введён в эксплуатацию на базовой кафедре Ассоциации документальной электросвязи (АДЭ) «Технологии электронного обмена данными» (ТЭОД) в МТУСИ. Это первое в России типовое решение для образования, полностью реализованное на отечественных технологиях, прошедшее апробацию и готовое к тиражированию.

В состав ТОМИИТ входят шестнадцать вычислительных комплексов Эльбрус 101 и Эльбрус 801, функционирующих над управлением отечественной операционной системы ОС Альт, два высокопроизводительных коммутатора Русьтелетех RTT A220-24-4G.

С целью расширения возможностей использования ТОМИИТ и улучшения практических навыков студентов в разработке и внедрения отечественных программно-аппаратных средств, реализующих современные интернет-технологии, разработаны перспективные направления доработки ТОМИИТ, которые рассматриваются в докладе:

- обеспечение удалённого доступа к ресурсам модуля с управлением идентификацией и полномочиями;
- переход на новую версию ОС Альт 9, который позволит получить доступ к новым функциям, утилитам, а также политикам безопасности;
- использование новых инфраструктурных вариантов построения, с учётом экономичности, функциональности и безопасности, на базе многопользовательских решений;
- создание межвузовского центра информационной безопасности.

Литература

[1] Кремер А. С., Мальянов С. А., Малюк А. А «Обеспечение доверия и безопасности при использовании ИКТ» Учебное пособие. — М.: ОГО АДЭ, 2017.

[2] Основы технологии сети Интернет: учебно-методическое пособие/МТУСИ; Кремер А. С., Иванюк А. В. —М.: ООО ФОП, 2019. — 200 с.

[3] Основы технологии сети Интернет: лабораторный практикум/МТУСИ; Кремер А. С., Иванюк А. В. — М.: ООО ФОП, 2019. — 376 с.

[4] Иванюк А. С. «Типовой отечественный модуль изучения интернет-технологий». — М.: Документальная электросвязь, №28, 2018

[5] Кирсанов В. А, Лопухов Р. С. «Апробация типового отечественного модуля изучения интернет-технологий»/Пятнадцатая конференция «Свободное программное обеспечение в высшей школе: Материалы конференции / Переславль, 7–9 февраля 2020 года. — М.: МАКС Пресс, 2020. — 180 с.

[6] Малиночкин В. С. «Лабораторный комплекс по изучению базовых сервисов сети Интернет с использование свободного программного обеспечения» /Пятнадцатая конференция «Свободное программное обеспечение в высшей школе: Материалы конференции / Переславль, 7–9 февраля 2020 года. — М.: МАКС Пресс, 2020. — 180 с.

Егор Орлов
Санкт-Петербург, Санкт-Петербургский политехнический университет Петра Великого (СПбПУ)

Программа профессиональной переподготовки ООО «Базальт СПО» и СПбПУ для системных администраторов ОС Альт

Обоснование

Декларируемый в России с 2014 года курс на технологическую независимость и импортозамещение поставил новые задачи перед ВУЗами и другими образовательными организациями при реализации программ высшего и дополнительного образования.

Необходимость развития национальной ИКТ-инфраструктуры с использованием отечественных технологий требует:

- адаптации программ высшего образования;
- активизации усилий по переобучению действующих ИТ-специалистов.

Для решения второй задачи необходимы программы профессиональной переподготовки.

При реализации Высшей Инженерной Школой СПбПУ программы АЛЬТСТАРТ для ИТ-специалистов государственных медицинских учереждений было выявлено значительное количество ИТ-специалистов, обладающих опытом и компетенцией в решении задач обеспечения работоспособности ИКТ-инфраструктуры медицинских организаций с использованием иностранных ИТ-продуктов. Тем не менее, при переходе на отечественное ПО в соответствии с требованиями 187-ФЗ, ИТ-специалисты сталкиваются с недостатком знаний и навыков, который не всегда может быть восполнен краткосрочными программами обучения.

Программа профессиональной переподготовки

Учитывая указанные тенденции и требования современного рынка труда в России, СПбПУ совместно с Базальт СПО запускает программу профессиональной переподготовки. Она направлена на подготовку/переобучение системных администраторов для использования решений Базальт СПО в своей профессиональной деятельности.

Программа профессиональной переподговоки "Системное и сетевое администрирование ОС Альт":

- 320 часов;
- 10 месяцев;
- базируется на авторизованных курсах Базальт СПО;
- дистанционный формат проведения;
- производственные практики под руководством сотрудников Базальт СПО и технологических партнёров.

Курсы программы:

- *ALTADM1*. Администрирование ОС Альт. Часть 1;
- *ALTADM2*. Администрирование ОС Альт. Часть 2;
- *ALTSHELL*. Автоматизация в ОС Альт;
- *ALTVIRT*. Виртуализация в ОС Альт;
- *ALTSERV*. Инфраструктурные службы в ОС Альт;
- *ALTINET*. Интернет-службы в ОС Альт;
- *ALTSEC*. Информационная безопасность в ОС Альт;
- *ALT-VKR*. Выпускная квалификационная работа.

Дмитрий Гудзенко
Москва, Центр компьютерного обучения «Специалист» при МГТУ им. Н. Э. Баумана

Специалист.Ру: кто учил Майкрософт, теперь учит Линукс

Аннотация

УЦ Специалист с 2001 года является лидером в подготовке системных администраторов. Статистика по обучаемым является хорошим «зеркалом» для анализа тенденций. В докладе будет показаны тренды изменения спроса на обучение по различным технологиям и продуктов разных вендоров за последние 20 лет.

В нашей стране в силу менталитета программное обеспечение с открытым исходным кодом очень популярно и приветствуется во многих рабочих средах. Люди до сих пор воспринимают open source ПО как, в первую очередь, доступное и бесплатное (хотя это по факту может быть не так), что влияет на популярность технологий.

В 2019 году был проведён опрос среди 950 ИТ-руководителей из четырёх регионов мира, которые знакомы с корпоративным open source и располагают как минимум одним процентом[1] Linux-систем в составе вычислительного парка организации.

Некоторые из выводов исследования:

- Стратегическая роль корпоративного open source усиливается, проприетарного ПО — снижается;
- Корпоративный open source и облачные вычисления[2] неразрывно связаны;
- ИТ-руководители выбирают корпоративный open source по причине более высокого качества этого ПО;
- Обеспечение безопасности[3] — область, где корпоративный open source применяется наиболее часто.

[1] https://www.tadviser.ru/index.php/Linux
[2] https://www.tadviser.ru/index.php/Облачные_вычисления
[3] https://www.tadviser.ru/index.php/ИБ

Мы отмечаем, что уже практически любому проприетарному ПО можно найти альтернативу из open source, поэтому обязательно учитываем это обстоятельство при проектировании линеек курсов: в ассортименте «Специалиста» можно найти курсы по и тем, и другим продуктам. За последние два года количество курсов по Linux в портфеле УЦ «Специалист» увеличилось на 13%, а количество слушателей — на 37%.

С усилением роли облачных технологий ИТ-руководители рассчитывают на открытый код в построении платформы. 83% опрошенных в исследовании 2019 года указали, что корпоративный open source во многом позволил их организациям воспользоваться преимуществами облачных архитектур. 63% опрошенных ИТ-руководителей располагают гибридной облачной инфраструктурой[4]. Из 37%, у которых её пока нет, 54% планируют внедрить её в течение 24 месяцев.

На наших курсах мы учим отталкиваться не от продукта, а от потребностей предприятия. Внедрение любого решения, если оно требует тонкой настройки и привлечения ресурсов со стороны, может потребовать немалых вложений. В СПО-решениях тоже есть уязвимости, поэтому при выборе решения надо руководствоваться скорее потребностями вашего бизнеса, например, параметрами производительности системы. Если security-ПО не выдержит нагрузки, то не имеет значения, какое оно — открытое или нет.

Если говорить о конкретных open source-продуктах, то есть ряд решений, которым мы доверяем и используем на наших курсах.

Это AlienVault OSSIM — система управления безопасностью, применяется в курсе защиты от хакерских атак. Мы показываем, как с её использованием проводить аудит безопасности сети и управлять инцидентами.

На курсах по этичному хакингу применяется и Nessus — один из самых популярных сканеров уязвимостей. Конечно, всё показываем, как применять на практике. К примеру, одна из практических работ построена на обучении использованию Nessus для инвентаризации уязвимостей компьютеров лабораторий.

В связи с трендом на импортозамещение в госсекторе, сейчас распространяются Unix-подобные системы, Linux-дистрибутивы, часто-

[4] `https://www.tadviser.ru/index.php/`Статья:Облачная_инфраструктура_(рынок_EMEA)

мизированные под отраслевые задачи. Среди них можно назвать решения Astra Linux, «Базальт СПО».

Как авторизованный центр обучения, мы можем отслеживать спрос как на сертифицированных специалистов по этим продуктам, там и в целом на широту распространения дистрибутивов. Судя по возрастающему спросу на пользовательские и админские курсы по Astra Linux и «Базальт СПО», востребованность этих систем 2-3 года назад стала ощутимо расти и продолжает набирать рост. Популярность этого направления в Центре «Специалист» выросла на 23% за последний год.

Есть динамика — процент используемого в компаниях проприетарного софта падает несколько последних лет.

Если оперировать нашими данными по админским курсам и сравнить их популярность по компаниям-поставщикам, то можно сказать, что Linux выигрывает у Microsoft.

Последнее время можно отследить следующую тенденцию: с распространением DevOps-методологии, инструментов контейнеризации Docker и Kubernetes, продвинутые курсы по Linux стали намного более востребованы.

Максим Авилов, Иван Хахаев
Санкт-Петербург, СПбГЭТУ «ЛЭТИ»

Инфраструктурные решения на основе «Альт Образование» в СПбГЭТУ «ЛЭТИ»

Аннотация

Обсуждается механизм обеспечения поддержки и обновления SSL-сертификатов от Let's Encript для сервисов, входящих в состав дистрибутива «Альт Образование». Приводится пример скрипта, обеспечивающего автоматическое обновление таких SSL-сертификатов для web-сервера Apache2 в дистрибутивах АЛЬТ.

В 2020 году на основе «АЛЬТ Образование» в СПбГЭТУ «ЛЭТИ» были развёрнуты два крупных сервиса — LMS Moodle[1] [1] и «облачное» хранилище NextCloud[2].

[1] `https://vec.etu.ru`
[2] `https://cloud.etu.ru`

Сервисы развёртывались на виртуальных машинах (VM) на кластере из 5 серверов [2] под управлением дистрибутива Proxmox VE 6.2.

Характеристики VM с LMS Moodle:

- ОС «АЛЬТ Образование» 9.1;
- количество зарегистрированных пользователей — около 14000;
- VM с дисковым пространством 4 Тб, RAM 64 Гб, 6 ядер 2,2 ГГц;
- среднее количество пользователей в день — около 2000;
- количество курсов — около 4300.

Характеристики VM с NextCloud:

- ОС «АЛЬТ Образование» 9.1;
- VM с дисковым пространством 20 Тб, RAM 32 Гб, 4 ядра 2,2 ГГц;
- квота на пользователя 10 Гбайт (рассчитано на 2000 пользователей — преподавателей и сотрудников).

Оба сервиса используют стек LAMP.

Одной из задач, решаемых при развёртывании указанных инфраструктурных решений, было обеспечение корректной работы https с подтверждением сертификата. Сертификаты, генерируемые при установке дистрибутива, не подтверждаются каким-либо центром сертификации, поэтому пользователи получают сообщение о вероятной угрозе безопасности.

Популярным решением является использование SSL-сертификатов от Let's Encrypt. Такое решение связано с бесплатным распространением, простотой получения и внедрения сертификатов, которые подтверждаются центром сертификации Let's Encrypt [3].

Однако на wiki сообщества АЛЬТ [4] даётся решение по использованию SSL-сертификатов от Let's Encrypt для веб-сервера NGINX, в то время как настройки web-сервера Apache2 в дистрибутивах АЛЬТ имеют нюансы, не позволяющие использовать информацию по настройке SSL-сертификатов от Let's Encrypt для Apache2 из других дистрибутивов.

Поскольку в настройках виртуального хоста по умолчанию для сервиса httpd2 в дистрибутивах «АЛЬТ Образование» настроены редиректы с http на https, подтверждение сервера (домена сайт), на котором расположен сайт, для обновление SSL становится проблематичным.

Это связано с тем, что в процессе обновления сертификата необходимо проверить права на домен. Из-за ограничений безопасности в корпоративной сети СПбГЭТУ «ЛЭТИ» метод обновления SSL через DNS-запросы не подходит, т. к. дополнительные TXT-записи с ключами не прописываются. Поэтому проверка происходит через предоставление HTTP-ресурса с определённым URI. Для корректного подтверждения прав на домен агент Certbot проводит тест-запросы с соответствующими ключами через 80 порт по HTTP [5].

В ситуациях, когда на виртуальном хосте настроены редиректы с HTTP на HTTPS, при обновлении сертификата возникает ошибка следующего вида:

```
Domain: aaa.bbb.ru
Type:   unauthorized
Detail: Invalid response from https://aaa.bbb.ru/.well-known/
acme-challenge/549OU8Q5XsKvzZ_I_QpLWS-V2gc0DrlKwH2oSDyWa1A
[xxx.xxx.xxx.xxx]: "<?xml version=\"1.0\" encoding=\"UTF-8\"?>\n
<!DOCTYPE html PUBLIC \"-//W3C//DTD XHTML 1.0 Strict//EN\"\n
\"http://www.w3.org/TR/xhtml1/D"
```

Получается, что центр сертификации не может подтвердить домен сайта, для которого требуются изменения информации по сертификату. Результатом возникшей проблемы является сообщение:
`To fix these errors, please make sure that your domain name was entered correctly and the DNS A/AAAA record(s) for that domain contain(s) the right IP address.`

Возможным решением данной проблемы является возврат к предоставлению информации по HTTP, но вручную каждые три месяца корректировать конфигурационные файлы виртуального хоста неэффективно. Поэтому можно автоматизировать процедуру обновления SSL при помощи скрипта, который должен выполнять следующие действия.

1. Убирать редиректы с HTTP на HTTPS в нужном конфигурационном файле виртуального хоста;
2. Проверять корректность синтаксиса конфигурационного файла виртуального хоста;
3. Запрашивать выдачу нового SSL-сертификата;
4. Восстанавливать убранные редиректы;

5. Перезапускать сервис httpd2 для использования обновлённых сертификатов.

Ниже приведён пример такого скрипта.

```
#!/bin/bash
sed -i "/^Rewrite/d" "/etc/httpd2/conf/sites-enabled/000-default.conf"
httpd2 -t 2> /root/script_tmp/certbot_httpd2.log
systemctl reload httpd2 2>> /root/script_tmp/certbot_httpd2.log
/usr/bin/certbot -q renew 2>> /root/script_tmp/certbot_httpd2.log
sed -i "/^</VirtualHost>/d" "/etc/httpd2/conf/sites-enabled/000-default.conf"
echo "RewriteEngine On" >> /etc/httpd2/conf/sites-enabled/000-default.conf
echo "RewriteCond %{HTTPS} !=on" >> /etc/httpd2/conf/sites-enabled/000-default.conf
echo "RewriteRule ^/(.*) https://%{HTTP_HOST}/$1 [R,L]" >> /etc/httpd2/conf/sites-
enabled/000-default.conf
echo "</VirtualHost>" >> /etc/httpd2/conf/sites-enabled/000-default.conf
tail -n8 /etc/httpd2/conf/sites-enabled/000-default.conf >> /root/script_tmp/certbot_
httpd2.log
systemctl restart httpd2 2>> /root/script_tmp/certbot_httpd2.log
```

Такой скрипт можно добавить в планировщик задач чтобы своевременно обновлять сертификаты необходимых сайтов. Успешные практические результаты были получены для указанных выше сервисов на основе Apache2 [6].

В случае обновления сертификата командой `certbot -q renew` сертификат не мог корректно обновиться, но при помощи приведённого выше скрипта обновление сертификата проходит успешно.

Литература

[1] *Хахаев И. А.* Снова об оценке электронных образовательных ресурсов // Свободное программное обеспечение в высшей школе : Сборник тезисов XV конференции, Переславль, 07–09 февраля 2020 года / Отв. редактор В.Л. Чёрный. — Переславль: ООО «МАКС Пресс», 2020. — С. 82–85.

[2] `https://linux-hardware.org/?probe=70139de021`

[3] Документация LetsEncrypt. `https://letsencrypt.org/ru/docs/`

[4] LetsEncrypt. `https://www.altlinux.org/LetsEncrypt`

[5] Как работает Let's Encrypt. `https://letsencrypt.org/ru/how-it-works/`

[6] The Apache HTTP Server Project. `https://httpd.apache.org/`

Иван Панченко
Москва, Postgres Professional

Чего ждёт, а чего не ждёт бизнес от ИТ-образования

О кадровой проблеме говорят сейчас все. Конкуренция за кадры обостряется, и коронавирусная пандемия усугубляет её, выравнивая условия в мировом масштабе.

Недостаток кадров вынуждает компании-разработчики обращаться к образованию, искать сотрудников среди вчерашних и сегодняшних студентов, участвуя в их подготовке и доучивании/переучивании.

Всё это должно привести к созданию экосистемы воспроизводства кадров, которое невозможно без получения образовательными организациями обратной связи от компаний.

Для устойчивости кадровых структур компаний в них должна быть полноценная пирамида кадров с системой внутреннего роста и самообновления. Однако на входе этой пирамиды — выпускники образовательной системы. Кто же нужен ИТ-индустрии?

Тут речь идёт не о уровнях сотрудников, которые обычно условно называются Junior/Middle/Senior, а об уровне подготовки входящих сотрудников. Назовём их условно «отличник, хорошист, троечник». Бизнесу нужны все они, в том или ином соотношении.

Как правило, в начале трудового пути сотрудник проходит обучение и адаптацию. Считается, что в течение испытательного срока (до 3 мес) он должен начать приносить пользу компании. Поэтому на первое место выходят требования к обучаемости сотрудников. Обучаемость определяется двумя компонентами — адаптивность мышления и базовые знания. Задача общего профессионального образования (вузы и ссузы) — формирование этих компонент. Они мало зависят от быстротечной конъюнктуры и поэтому идеально подходят для образования. Не так важно, чему учить, лишь бы это формировало «мозги» обучающихся. К предметам, обладающим такими свойствами, относятся: классическая математика («приводит ум в порядок»), решение задач по физике (учит построению мысленных моделей реального мира), классическое программирование и дискретная математика (даёт систематический подход к профессии). Как минимум половина учебного времени должна отдаваться на «формирование мозгов», а не на усваивание конкретных навыков. В этом смысле, снижается ценность

классических лекций, гораздо более важным являются практические занятия во взаимодействии с преподавателем, по возможности индивидуальные.

Нет смысла загружать головы практическими навыками, которые быстро устареют и не будут востребованы. Изучивший питон вряд ли легко перейдёт на C или даже Java. Наоборот — перейдёт легко. ВУЗы должны концентрироваться на фундаментальном и сложном. Обучение практическим навыкам — дело учебных центров.

Кадровый голод второго порядка описывается фразой «некому учить». Действительно, преподавателей, способных на что-то, не хватает. Поэтому имеется задача подготовки преподавателей. Это государственная задача и мы ждём поддержки её решения от государства. Однако и сами компаниям следует самоорганизовываться и действовать совместно, делегируя сотрудников для отбора и обучения тех, кто будет учить других.

Дмитрий Мазуркевич, Юлия Перязева, Иван Хахаев
Санкт-Петербург, СПбГЭТУ «ЛЭТИ»

Проект: Adaptive Learning Toolbox
https://github.com/kalganov/moodle-lti-bridge#adaptive-learning-toolbox

Применение IRT для анализа качества тестовых материалов в LMS Moodle

Аннотация

Работа посвящена решению задачи анализа качества тестовых материалов в LMS Moodle на основе Item Response Theory (IRT), а именно модели Г. Раша. Приложение реализовано на Python, фреймворк Django. Для интеграции приложения в LMS Moodle используется протокол LTI.

Введение

Активное внедрение дистанционной и смешанной форм обучения делают компьютерное тестирование одним из основных инструментов педагогических измерений. Правильно построенные онлайн и смешанные курсы могут быть так же эффективны, как и традиционное

очное обучение [1], но для этого, прежде всего, необходимы качественные диагностические материалы, в частности, тесты. Тесты должны обладать необходимыми статистическими характеристиками и обеспечивать высокое качество измерений, с учётом того, что оценивание в образовании осложняется латентным характером измеряемых переменных. Существуют классическая и современная теория тестирования Item Response Theory (IRT). Классическая теория имеет ряд принципиальных недостатков. В частности, тестовые баллы зависят от трудности заданий, а трудность задания зависит от выборки тестируемых, имеет место нелинейность тестовых баллов испытуемых. IRT позволяет анализировать качественные данные с помощью количественных методов. Мера измерения параметров, в частности, у модели Раша является линейной, оценка трудности тестовых заданий не зависит от выборки испытуемых, оценка уровня подготовленности испытуемых не зависит от используемого набора тестовых заданий [2].

Широко используемая в образовании LMS Moodle содержит развитую тестовую подсистему с встроенным механизмом автоматизированной статистической обработки результатов тестирования и вычисления показателей качества тестовых материалов на основе классической теории тестирования. Несмотря на ряд работ посвящённых статистическому анализу тестов и, в частности, более эффективной IRT для анализа и повышения качества тестов в LMS Moodle [3–5], готового решения для статистической обработки тестовых заданий на основе IRT в Moodle не было найдено.

Цель работы — разработка приложения для анализа качества тестовых материалов в LMS Moodle на основе наиболее простой и эффективной модели IRT, модели Георга Раша.

Item Response Theory

Математическая модель Item Response Theory определяется следующим образом:

$$P(\theta) = c + \frac{1-c}{1+e^{-a(\theta-b)}},$$

где P — вероятность решить задание со сложностью b, дискриминацией a, и угадыванием c при уровне подготовленности обучающегося θ. Дискриминация определяет насколько задание способно различать студентов с разным уровнем θ. IRT позволяет оценить латентный параметр подготовленности обучающегося, исходя из предположения,

что и он и сложность задания размещены на одной шкале и измеряются в одних и тех же единицах — логитах [2]. Такую модель также называют трёх параметрической логистической моделью, а также Item Response Function. На практике широкое распространение получила упрощённая модель Георга Раша:

$$P(\theta) = \frac{e^{1.7(\theta - b)}}{1 + e^{1.7(\theta - b)}}$$

Такую модель также называют однопараметрической моделью IRT. Преимуществами модели Раша является простота её использования, так как для заданий необходимо определить только b, и способность данной модели отделить оценки испытуемых от оценки трудности задания.

Приложение анализа качества тестовых материалов в LMS Moodle

Предлагаемое приложение разработано на языке Python (фреймворк Django), интегрируется в LMS как внешнее приложение по протоколу LTI, доступно в курсах преподавателям для анализа тестовых материалов.

В приложении реализованы следующие этапы анализа тестов в рамках модели Раша: оценивание параметров, исследование согласия эмпирических данных с моделью, анализ заданий теста и испытуемых, характеристических кривых для заданий теста (Рис. 1).

Параметры сложности тестовых заданий и подготовленности обучающихся рассчитывается из эмпирических данных [3].

В приложении строятся характеристические кривые для заданий теста и проводится сравнение этих теоретических кривых с экспериментальными данными, исследуется согласие эмпирических данных с моделью и, следовательно, качество задний. На рисунке 2 видно, что первое задание расходится с моделью Раша, это может объясняться как несовершенством задания по форме или по содержанию, так и ошибками в организации процедуры тестирования. Второе задание соответствует модели.

Приложение строит характеристические кривые для заданий теста. Например, из приведённых графиков (Рис. 3) видно, что задания неравномерно покрывают требуемый диапазон уровней подготовлен-

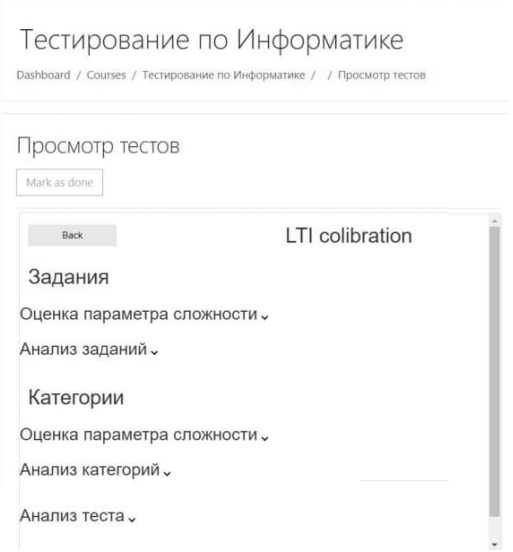


Рис. 1: Выбор этапа анализа теста

ности испытуемых, а часть заданий дублируют друг друга по сложности, если возможно по содержанию, то часть из них можно удалить из теста без ущерба его общей дифференцирующей способности.

Заключение

Разработанное приложение будет полезно для разработки качественных тестирующих материалов в LMS Moodle. Качество тестовых заданий в приложении оценивается в рамках модели Г. Раша IRT, предполагается расширение приложение другими моделями.

Литература

[1] Chirikov, Igor, et al. , Online education platforms scale college STEM instruction with equivalent learning outcomes at lower cost., 2020

[2] Ким В. С., Тестирование учебных достижений. Монография., 2007

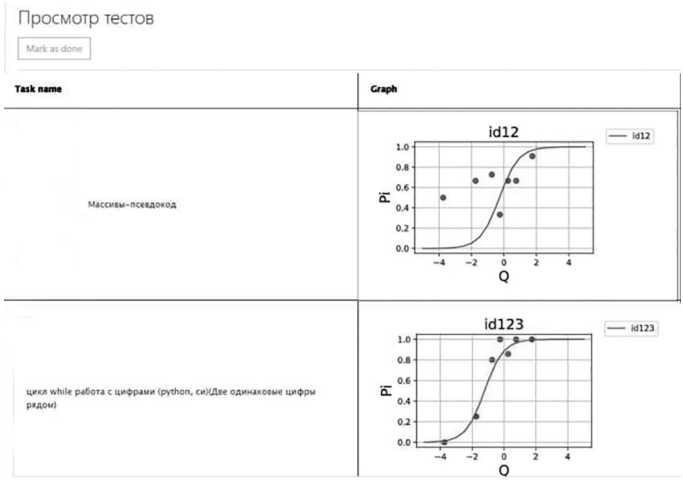


Рис. 2: Характеристические кривые заданий

[3] Протасова И. В., Толстобров А. П., Коржик И. А., Методика анализа и повышения качества тестов в системе электронного обучения Moodle, 2014

[4] Савкина А. В., Нуштаева А. В., Савинов И. А., Вечканова Ю. С. , Статистические исследования качества электронных образовательных ресурсов, 2019

[5] Юбко А. А., Ефромеева Е. В., Жаров В. К., Анализ статистики прохождения теста в электронной образовательной среде Moodle, 2019

[6] Челышкова М. Б., Теория и практика конструирования педагогических тестов: Учебное пособие., 2002

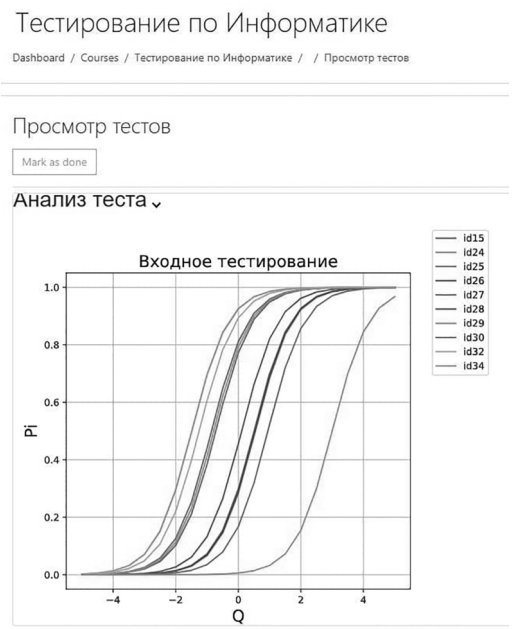


Рис. 3: Характеристические кривые заданий теста

Тамара Пестунова, Владимир Шумейко
Новосибирск, ООО «Специальные технологии защиты информации», НГУ

Опыт реализации образовательных проектов в сфере свободного программного обеспечения для целей импортозамещения на примере ОС «Альт»

Аннотация

В работе рассматриваются концептуальные аспекты обучения ИТ-специалистов, осваивающих Linux-системы в целях перехода на отечественное системное программное обеспечение. На основе опыта авторизованных курсов по администрированию ОС «Альт» анализируются некоторые методические особенности учебного процесса, обусловленные потребностями слушателей в приобретении за ограниченное время

основных компетенций для миграции ИТ-инфраструктуры с платформы Windows на Linux.

Переход на отечественное *программное обеспечение* (далее — ПО) в современных условиях является актуальной задачей для государственных и многих коммерческих компаний. В части системного ПО наиболее ярко это выражается в необходимости перехода с операционной системы (далее — ОС) Windows на ОС, разработанные российскими компаниями на платформе ядра Linux.

Трудности при реализации подобных проектов обусловлены рядом причин, среди которых значимую роль играет недостаток квалифицированных ИТ-кадров, способных грамотно внедрять и администрировать ОС семейства Linux, используя широкий спектр их функциональных возможностей и механизмов обеспечения безопасности информации. Несмотря на неизменно повышенный интерес к Linux-системам, они долгое время остаются уделом узкой прослойки специалистов, порождая в среде пользователей массу «страшилок» и мифов о трудности работы с ней. При этом активная маркетинговая политика Microsoft, которая с начала 2000-х годов предоставляла образовательным организациям бесплатно весь комплекс своих продуктов для разработчиков ПО и системных администраторов, способствовала тому, что из образовательных программ большинства вузов за короткое время исчезло всё другое системное ПО, а подготовка специалистов по Linux-системам стала своеобразным эксклюзивом. А поскольку с учётом сертификации по требованиям безопасности в соответствии с нормативными документами ФСТЭК России, системное и прикладное ПО Майкрософт стало основой государственных и муниципальных информационных систем, информационных систем персональных данных, объектов критической инфраструктуры, то потребности в освоении Linux-платформ у обслуживающих их ИТ-специалистов в процессе трудовой деятельности практически не возникало.

С учётом сказанного основная проблема при внедрении и изучении системы Linux в целом и «Альт» в частности обусловлена инертностью мышления системных администраторов и обычных пользователей, привыкших к ПО от Майкрософт. Но в рамках проектов миграции на Linux-платформы задача приобретения ими необходимых компетенций должна эффективно решаться в сжатый период времени. В этой ситуации важно дать понять пользователям, что ОС Linux (в дальнейшем будем рассматривать именно ОС Альт), ничуть не ху-

же ОС Windows. Она просто другая. Никто ведь не требует от ОС Android или iOS быть похожими на Windows. Основываясь на опыте преподавания, можно выделить следующие вопросы, возникающие при первом знакомстве с ОС Альт:

- непонимание организации структуры хранения файлов (первый вопрос: «А где диск С?»).
- неприятие интерфейса («интерфейс из 90-х»), сюда же можно отнести и непривычные изображения указателя мыши.
- неприятие работы с командной строкой («А можно это графической утилитой сделать?», «возврат в 90-е»).

Очевидно, что они связаны исключительно с субъективным восприятием ОС, и не относятся к глобальным недостаткам системы. Успешность преодоления подобных психологических проблем в процессе обучения во многом зависит от преподавателя, его опыта работы с ОС семейства Linux и личной харизмы.

Технически проведение курсов по ОС Альт не представляет проблем — ОС легко устанавливается и практически не требует какого-либо специфического программного или аппаратного обеспечения. Целесообразно использовать технологию *виртуальных машин* (далее — ВМ), что позволяет гибко подходить к организации курса и исключить возможные потери времени на устранение неполадок, возникающих от неграмотных или преднамеренных действий пользователей. При этом не так важно, на базе какой ОС устанавлены ВМ – это может быть и ОС Windows, и ОС Linux, хотя технологически при большом потоке слушателей последние будут более устойчивы, и обслуживание их будет более лёгким. Но это при условии, что персонал, администрирующий учебную инфраструктуру является достаточно квалифицированным в ОС Linux. ВМ позволят использовать технологии snapshot'ов — возможности делать снимки состояний ВМ для их последующего восстановления в случае, если что-то пойдёт не так. Это самое востребованное их свойство в процессе проведения курсов, ибо пользователи могут неграмотными действиями не только полностью приводить систему в негодность, но и ставить в тупик преподавателя.

Подобным образом организованы курсы по администрированию ОС Альт в авторизованном учебном центре ООО «Спецтехнологии». В 2021 году проводилось корпоративное обучение трёх групп численностью от 6 до 8 слушателей, являющихся специалистами подразделе-

ний технического сопровождения и администрирования информационных систем государственного учреждения. Концепция авторизованных курсов предполагает изучение основ применения и администрирования готовой ОС Альт в базовом курсе (часть 1), а затем более глубокое освоение системы (включая установку и детальное конфигурирование) в расширенном курсе (часть 2). В ходе курсов сложилось впечатление, что распределение тематики на эти две части не является оптимальным и порождает примерно одинаковые вопросы у преподавателей и слушателей.

Самый популярный вопрос — «почему установка ОС вынесена в расширенный курс» — вполне логичен: ведь если пользователь собрался изучить ОС Альт, её надо для начала поставить. Тем более, что в установке нет ничего сложного, хотя редактор разделов в ОС Альт, по-прежнему оставляет желать лучшего. Ещё в 4-й версии отзывы по установке были не утешительны: «*Процесс инсталляции в ALT Linux 4.1 Desktop проходит не безоблачно. Во-первых, желательно заранее иметь подготовленные разделы на жёстком диске или, как минимум, необходимо обеспечить свободное место. Установщик не умеет менять размеры разделов без потери в них данных. Вторая возможная проблема — это определение видеокарты. Здесь творится какой-то хаос. Конфликт драйверов, неверное разрешение, неверная глубина цвета. Впрочем, и здесь нет ничего фатального. Всё исправляется вручную, уже после установки системы* [1]. »

Сейчас таких глобальных проблем не наблюдается, но работа редактора разделов до сих пор вызывает вопросы. Например, при создании таблицы разделов на пустом диске, сразу создаётся раздел размером во всё дисковое пространство, что как-то не логично и в целом не нужно. Приходится по нескольку раз править разделы и добиваться нужной конфигурации разделов. Есть вопросы и к организации хранения логов в системе и к системе установки пакетов. Понятно, что это особенности конкретной системы, но... Ведь и справочной информации по ОС Альт наблюдается сильный дефицит. Если уже делать что-то отличное от других, надо и справочную систему подтянуть. Очень грустно смотреть на практически пустую альт-вики страницу по адресу `altlinux.org`.

Второй камень преткновения в списке тем — скрипты. Многие пользователи оказались настолько далеки от программирования, что

[1] `https://www.ixbt.com/soft/alt-linux-41-desktop-1.shtml`

такие задачи вызывают у них неподдельный ужас — это обстоятельство можно как раз отнести к числу «плодов» массовой подготовки системных администраторов исключительно на ПО от Microsoft, о чём было написано выше. Но при изучении любых Linux-систем без этой темы курсы становятся неполноценными. Решением может стать отдельный курс по ОС Альт для «начинающих», где вопросы написания скриптов будут отсутствовать, а далее уже по мере приобретения навыков работы в системе, добавлять основы программирования под Linux при увеличенном объёме часов на данную тему. И такое решение в настоящий момент уже присутствует в некотором виде. Но, с другой стороны, при планировании обучения, важно обращать внимание на то, чтобы в группах были слушатели примерно с одинаковым уровнем исходной подготовки к работе в Linux, поскольку при проведении корпоративного обучения состав каждой группы часто определяется заказчиком из других соображений.

В целом же можно сказать, что ОС Альт вполне работоспособна и интересна, как основа для обучения администрированию и применению ОС семейства Linux, так и в роли платформы рабочей среды корпоративных информационных систем.

Александр Андреев, Виктор Яковлев
Москва, Московский Физико-Технический Институт (НИУ)
https://github.com/carzil/hellos

Учебная операционная система HellOS

Аннотация

Для того, чтобы разобраться во внутреннем устройстве операционных систем, необходимо исследовать их исходные тексты, а ещё лучше — дописать или переписать их отдельные компоненты. Для этих целей открытость исходного кода является обязательным требованием к операционным системам, но открытый код ещё не делает систему доступной для изучения.

Современные системы общего назначения, за десятилетия своего развития, превратились из игрушечных концептов в проекты с огромным количеством исходного кода, который необозрим в рамках любого курса. Существуют специализированные решения для образовательных целей, но их реализации не всегда соответствуют современным представлениям о качестве кода.

В работе представлена «игрушечная» операционная система, которая не предназначена для промышленного применения, но при этом является удобной для изучения внутреннего устройства ядра. Предложенная операционная система была использована в рамках семестрового курса по разработке операционных систем в МФТИ.

О курсе по устройству операционных систем

Классическим курсом по внутреннему устройству операционных систем является курс Operating Systems Engineering американского университета MIT. Методически курс оформлен в виде «лабораторных работ», в рамках выполнения которых студенты должны реализовать какую-либо функциональность операционной системы. При этом студенты не пишут всю операционную систему с нуля, а имеют некоторый «скелет», — минимально функциональную систему небольшого размера. В рамках оригинального курса используется операционная система XV6, — открытая операционная система для архитектуры x86, которая является клоном очень старой оригинальной UNIX 6-й редакции. В России похожий курс читается в Школе анализа данных от Яндекс, где также используется система XV6.

В 2020 году на направлении «Информатика и вычислительная техника» ФПМИ МФТИ прошла первая итерация курса по внутреннему устройству операционных систем, идейно повторяющий курсы в MIT и ШАД. В рамках этого курса были изучены: процесс загрузки ОС, прерывания процессора, виртуальная память, планировщик процессов, работа с HDD в различных режимах, базовые принципы работы файловых систем, изоляция процессов. В отличии от курсов-прообразов, для курса в МФТИ была разработана своя базовая операционная система.

Почему не XV6

Использовать GNU/Linux или существующие BSD-решения общего назначения в учебных целях невозможно, — одного семестра просто не хватит на то, чтобы изучить архитектуру минимальной рабочей системы. Существуют системы, существенно более компактные, хотя и ограниченные в функциональности. Одной из таких систем является XV6 [1], разработанная в 2006 году в MIT специально для учебного курса по низкоуровневому программированию.

Из преимуществ этой системы можно отметить наличие готового учебного курса (методических материалов), который можно использовать с минимальными переработками. Учебные материалы, как и сама операционная система, опубликованы в открытом доступе. При этом, качество исходных текстов самой системы оставляет желать лучшего: в коде отсутствует единый стиль кода, часто используются однобуквенные переменные, файлы userspace-программ перемешаны с файлами ядра. Также в её коде используются устаревшие механизмы: не используется ACPI, нет поддержки DMA для чтения с жёстких дисков и т. д.

Мы ставили перед собой цель не сделать ещё один курс по XV6, а дать дополнительный современный материал сверх того, что мотивированные студенты могут найти в открытом доступе, поэтому для этого потребовалась новая операционная система.

Архитектура системы HellOS

Система HellOS реализована на языке C для архитектуры x86 и предназначена для работы в виртуальной машине QEMU. Представляет собой простое монолитное ядро, расположенное в верхних адресах памяти. Поддерживает загрузку с помощью multiboot-совместимого загрузчика (подходят все современные загрузчики ОС). В HellOS реализованы базовые функции для работы с примитивами x86: обработка вектора прерываний, работа с виртуальной памятью и т.д. Реализованы функции парсинга и валидации ACPI таблиц. Не используется устаревший PIC, вместо него для управление прерываниями используется APIC/LAPIC. HellOS имеет базовый не fair-share планировщик процессов, не поддерживающий 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 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. Реализация виртуальной памяти в x86.

4. Планировщики процессов.
5. Работа с HDD (режимы PIO и DMA).
6. Базовые принципы работы файловых систем на примере ext2.
7. Запуск userspace-приложений.

Лабораторные работы, предлагаемые студентам

Некоторые части операционной системы намеренно остались не реализованными для того, чтобы студенты самостоятельно могли их сделать при выполнении лабораторных работ.

1. Подготовка окружения (настройка кросс-компилятора, qemu, gdb, make etc). Это самая простая, но самая важная лабораторная работа. В ходе неё студенты устанавливают нужны утилиты и программы для компиляции, разработки и тестирования кода, которые они будут писать в последующих работах.
2. Калибровка APIC таймера и разработка драйвера PS/2-совместимой клавиатуры. Эта работа учит студентов работе с примитивами x86 (I/O операции процессора, прерывания, и т.д.). А также учит студентов искать и использовать документацию по x86.
3. Менеджер виртуальной памяти. Как показала первая итерация курса, это была самая сложная лабораторная работа. Она предлагает студентам реализовать базовый аллокатор физических фреймов, а поверх него средства для аллокации виртуальной памяти.
4. Реализация механизма переключения контекста и изоляция процессов. Базовый context switch поддержан в HellOS, однако обладает рядом недостатков: например, не поддерживается FPU-контекст (x87 и SSE), отсутствует изоляция виртуальных адресных пространств для различных процессов. Эти недостатки и предлагается исправить студентам.

Результаты и дальнейшее развитие

Реализована операционная система, которая опубликована на GitHub [2] под лицензией MIT. Первая апробация курса прошла осенью 2020 года, следующая итерация запланирована на осенний семестр 2021 года. В рамках первой итерации студенты смогли освоить

большинство тем, однако при этом у них возникли сложности с усвоением тем по виртуальной памяти.

В следующей итерации курса и HellOS будет добавлена поддержка архитектуры ARM, которая будет изучаться параллельно с x86.

Литература

[1] Xv6, a simple Unix-like teaching operating system, `https://pdos.csail.mit.edu/6.828/2020/xv6.html`

[2] Simple Operating System, `https://github.com/carzil/hellos`

Андрей Черепанов
Москва, ООО «Базальт СПО»
Проект: Sisyphus altlinux.org

Развитие дистрибутива Альт Образование: от 9.0 к 9.2

Аннотация

В докладе рассматриваются основные изменения в образовательном дистрибутиве Альт Образование с версии 9.0 по 9.2.

Новшества

1. Новые профили установки:

- Базовый образовательный комплект (минимальный достаточный набор для соблюдения требований к составу программного обеспечения по Распоряжению Правительства Российской Федерации от 18.10.2007 г. № 1447-р;
- Сервер видеоконференций (на базе Jitsi-Meet);

Возможность отмены ранее обязательной образовательной компоненты позволяет устанавливать систему для АХЧ без обучающего программного обеспечения.

2. Доустановка профилей после установки Альт Образование или иного дистрибутива Альт в виде метапакетов:

- task-edu (базовый образовательный комплект);

- task-edu-gradeschool (начальная школа);
- task-edu-highschool (средняя школа);
- task-edu-kde5 (среда KDE5);
- task-edu-preschool (дошкольное образование);
- task-edu-school (образовательное программное обеспечение для школ всех классов);
- task-edu-secondary-vocational (среднее профессиональное образование);
- task-edu-server-apps (серверные приложения);
- task-edu-teacher (программное обеспечение для учителей);
- task-edu-university (высшее образование);
- task-edu-video-conferencing (сервер видеоконференций).

3. Добавлен агент групповых политик (`https://www.altlinux.org/`Групповые_политики);

4. Образ для aarch64 работает на компьютерах с процессором Байкал-М: TF307-MB-S-D с прошивкой из SDK-M 4.4 и на моноблоке «Эдельвейс».

5. Переход на более новые ядра un-def (5.10) для поддержки нового оборудования.

6. В профиль для начальной школы добавлен trikStudioJunior.

7. Для разработчиков на Python добавлен pip для Python 2.x и Python 3.x.

8. Поддержка интерактивных досок: в профиль для учителя ставится среда OpenBoard, для настройки касаний — touchegg.

9. В части централизованного управления классом устаревший iTalc3 заменён на Veyon.

10. Добавлены пакеты alterator-control, alterator-grub и alterator-update-kernel для настройки загрузчика и обновления ядра в установленной системе.

11. Оптимизирована работа на aarch64, в том числе по умолчанию в качестве видеоплеера предлагается VLC, а не smplayer.

12. Программа локального разворачивания ролей deploy на базе Ansible (пока поддерживается предустановленная роль разворачивания PostgreSQL).

13. Вместо устаревшего freemind используется freeplane, вместо scratch — scratch-desktop.

Перспективные изменения

1. Добавление в Moodle системы автоматизированной проверки кода: Virtual Programming Lab или CodeRunner.

2. Восстановление в меню нелокализованного, но популярного Idle как IDE для разработки на Python.

3. Продолжение экспериментов по системе развёртывания систем из пакетной базы с настройкой и самопроверкой. Возможно расширение его для Moodle, Mediawiki, Nextcloud и КриптоПро.

4. Графический интерфейс для epm play (например, для установки VS Code или Microsoft Teams).

Яков Зайдельман[1], Анатолий Кушниренко[1], Александр Леонов[1,2], Миля Райко[1]
Москва, [1]ФГУ ФНЦ НИИСИ РАН, [2]МПГУ

Семейство отечественных свободно распространяемых цифровых образовательных сред ПиктоМир — ПиктоМир-К — КуМир и опыт его использования

Аннотация

В докладе будет описано семейство свободно распространяемых многоплатформенных цифровых образовательных сред (далее ЦОС), оптимизированных для преподавания азов программирования обучаемым всех возрастов, начиная с 6+, включая дошкольников, школьников, студентов педагогических университетов, воспитателей детских садов и учителей начальной и основной школы.

Это семейство призвано поддержать подход, при котором первые шаги в освоении программирования, независимо от возраста обучаемого, делаются в бестекстовой, пиктограммной среде составления программ управления виртуальными и реальными роботами. На первом этапе, доступном для обучаемых возраста 6+, осваиваются основные программные конструкции: следование, ветвление, повторение, подпрограмма (с однобуквенным именем), предусмотренные действующим ФГОС ООО. На следующем втором этапе более сложные программы управления освоенными ранее роботами составляются, по выбору обучаемого, в пиктограммной или в блочно-текстовой форме, с

возможностью использования в последнем случае целочисленных переменных и логических выражений. Лишь на последнем третьем этапе, накопив опыт составления и отладки пары сотен программ, обучаемые переходят в традиционную полнотекстовую среду программирования для составления в ней традиционных программ обработки числовой и текстовой информации.

Коротко о членах семейства ЦОС в порядке их появления на свет

КуМир. Полнотекстовая учебная среда програмирования КуМир — Комплект Учебных Миров — поддерживает придуманый академиком А. П. Ершовым в 1985 году паскалеподобный Школьный Алгоритмический Язык. Среда была реализована совместными усилиями лаборатории вычислительных методов мехмата МГУ и временного коллектива «Школа-1» АН СССР в 1985 году. Начиная с 1988 года и до наших дней КуМир широко используется в системе школьного образования России. Текущая версия КуМира может запускаться под управлением операционных систем семейств Linux, Microsoft и MAC OS и в ней могут успешно выполняться программы из учебников прошлого века. То есть, по выражению президента России Владимира Путина, пользуясь КуМиром «школьники всё ещё изучают языки, элементы программирования, которые применялись даже в прошлом веке» [1]. Кроме использования проверенных временем классических понятий программирования популярность КуМиру добавляют ещё одно обстоятельство. Ключевые слова, имена переменных и имена функций в КуМире можно писать на русском языке, используя буквы действующего ныне русского алфавита, введённого стандартом 1918 года. ЦОС КуМир — полнотекстовая среда программирования:

— обучаемый видит программу, как текст, состоящий из отдельных символов;

— обучаемый вводит программу, пользуясь клавиатурой, посимвольно;

— КуМир позволяет обучаемому вводить синтаксически неверную программы, что вынуждает обучаемого разбираться в диагностических сообщениях и учиться исправлять синтаксические ошибки.

Хотя Кумир снабжён замечательной системой мгновенной диагностики синтаксических ошибок и оригинальной системой отладки, он остаётся полнотекстовой системой, нацеленной на оптимизацию процесса решения достаточно сложных задач достаточно подготовленными обучаемыми. Использование КуМира, как и любой другой полнотектовой среды программирования, для решения простейших задач оказываетя совершенно неэффективным.

Грубо говоря, на начальном этапе обучения программированию новичков любого возраста нужна система с автоматической проверкой, позволяющая педагогу за 10 занятий добиться от каждого из учеников отладки 100 учебных программ. Ни одна полнотекстовая среда программирования такой возможности не даёт.

Отступление. Освоение азов программирования — трудная задача. Её сложность соизмерима со сложностью освоения азов чтения, письма и счёта. Для обучения традиционной грамотности в мире и в России имеется несколько устоявшихся традиционных методик. Задача обучения азам программирования исторически очень молода. И устоявшихся методик пока нет.

Пиктомир. Бестекстовая пиктограмная учебная среда программирования ПиктоМир была создана по заказу Академии наук РФ в отделе учебной информатики НИИСИ РАН в 2010 году [2]. ПиктоМир — свободно распространяемая программная система для изучения азов программирования дошкольниками и младшими школьниками. ПиктоМир позволяет ребёнку «собрать» из пиктограмм на экране компьютера (или из кубиков с наклеенными на грани пиктограммами команд) несложную программу, управляющую виртуальным или реальным исполнителем-роботом. ПиктоМир в первую очередь ориентирован на дошкольников, ещё не умеющих писать или на младшеклассников, не очень любящих писать, но неожиданно оказался весьма эффективным при обучении новичков любых возрастов [3].

Система может быть скачана с сайта ФГУ ФНЦ НИИСИ РАН для планшетов с тачскрином и лаптопов с операционными системами Windows, Android или MacOS X, или использована онлайн через веб-интерфейс. Допускается использование системы ПиктоМир и методических материалов в любых целях, в том числе и в коммерческих.

ПиктоМир быстро нашёл своё место в системе дополнительного, в том числе и платного, образования РФ в детских садах и начальных школах, используется в сотнях образовательных организаций. В комплект загрузки Пиктомира входят 45 практикумов, каждый из ко-

торых расчитан на одно получасовое занятие и насчитывает от 5 до 10 заданий, а также две методички объёмом несколько сот страниц, описывающие годовой курс «Алгоритмика для дошколят» для учащихся подготовительных групп детских садов и полугодовое продолжение этого курса для первоклассников.

ПиктоМир-К. Наш опыт стыковки в одном курсе ПиктоМира и КуМира показал, что переход от бестекстового программирования к текстовому сложен для обучаемых любого возраста и уровня подготовки. Одинаково сложен и для младшеклассников и для студентов педагогических университетов. Потому нами была разработана свободно распространяемая система блочно-текстового программирования ПиктоМир-К (ПиктоМир плюс Кумир) [4]. Эта система полутекстовая, построенная по тому же принципу, что и Scratch: ввод команд роботов и программных конструкций бесклавиатурный, пиктограммный, блочный, с блокировкой ввода синтаксически неправильных конструкций, что обеспечивает преемственность при переходе из ПиктоМира. А представление программы текстовое, на школьном алгоритмическом языке, что обеспечивает преемственность при переходе в КуМир. ПиктоМир-К свободно распространяем. Система может быть скачана с сайта ФГУ ФНЦ НИИСИ РАН для планшетов с тачскрином и лаптопов с операционными системами Android, MacOS X или Windows. Система может также использоваться онлайн через вебинтерфейс. Допускается использование системы ПиктоМир-К и методических материалов в любых целях, в том числе и в коммерческих.

Годовой вводный курс программирования в Институте детства МПГУ

Мы читаем этот курс в течение 5 лет. На первых порах мы использовали системы Пиктомир и КуМир и решение задач в КуМире вызывало определённые затруднения. В текущем учебном году мы используем семейство из трёх сред программирования: ПиктоМир $\longrightarrow$ ПиктоМир-К $\longrightarrow$ КуМир. Студенты получают 405 заданий по составлению программ. Из них

- 130 заданий в системе ПиктоМир,
- 170 заданий в системе ПиктоМир-К,
- 105 заданий в системе КуМир.

Для получения зачёта по каждой теме студент должен сдать 80% задач, то есть всего за год сдать более 300 задач. При такой схеме все обучаемые справляются с заданиями в системе КуМир гораздо легче, чем раньше. В частности, большинство студентов успевает выполнить зачётный объём заданий в часы занятий.

Предложения по преподаванию программирования в школах РФ

По нашему мнению, описанные выше первые два этапа изучения азов программирования следует перенести в начальную школу и включить в обязательную программу. В результате все выпускники начальной школы получат опыт составления нескольких сотен программ с использованием основных программных конструкций, предусмотренных действующим сегодня ФГОС ООО. Переход к полнотекстовой среде программирования можно будет провести в 5–6 классах основной школы.

Литература

[1] Стенограмма основной дискуссии конференции по искусственному интеллекту Artificial Intelligence Journey (AI Journey 2020) на тему «Искусственный интеллект — главная технология XXI века». Москва, 4 декабря 2020 URL: `http://kremlin.ru/events/president/news/64545`

[2] Rogozhkina, I., Kushnirenko, A, «PiktoMir: teaching programming concepts to preschoolers with a new tutorial environment.» Procedia — Social and Behavioral Sciences, 2011. Vol. 28. pp. 601–605. doi: 10.1016/j.sbspro.2011.11.114.

[3] Besshaposhnikov, N. O., Kushnirenko, A. G., Leonov. A. G.: Pictomir: how and why do we teach textless programming for preschoolers, first graders and students of pedagogical universities. In: Proceedings of the 13th Central & Eastern European Software Engineering Conference in Russia (CEE-SECR '17), Article No. 21, pp. 1–7. ACM, New York, NY, USA (2017).

[4] Бесшапошников Н. О., Кушниренко А. Г., Леонов А. Г., Малый А. А.: Проект двуязыковой пиктограммно-текстовой среды программирования ПиктоМир-К, Сборник тезисов Четырнадцатой конференции «СВОБОДНОЕ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ В ВЫСШЕЙ ШКОЛЕ», Переславль, 25–27 января 2019, Издательство ООО "МАКС Пресс" (Москва), стр. 64–65.

Иван Пономарёв, Олег Ивченко, Илья Селиванов, Нурас Ногаев, Иван Архипов
Москва, Московский физико-технический институт (государственный университет)

Проект: LJV (Lightweight Java Visualizer)
`https://github.com/atp-mipt/ljv`

Простая визуализация структур данных в Java

Аннотация

LJV (Lightweight Java Visualizer) — инструмент визуализации структур данных Java с использованием Reflection API для прохода по объекту и GraphViz для отрисовки. Проект изначально был разработан Джоном Хамером (John Hamer) в 2004 году на Java 1.4 под лицензией GPL, но с тех пор не обновлялся. Нами проект был переписан на современную версию языка Java, выложен на GitHub, а также добавлены новые фичи в виде дополнительных настроек отрисовки графов и более удобным интерфейсом для пользователя. LJV можно использовать для реверс-инжиниринга, а также он будет удобен преподавателям и студентам, чтобы изучать алгоритмы и структуры данных

История проекта

В 2004 году Джон Хаммер, который работал в Новой Зеландии, разработал инструмент для обучения студентов, чтобы они лучше освоили структуры данных. Долгое время, инструмент был написан на Java 1.4 и запускал заранее предустановленный на компьютере GraphViz. Но в осенью 2020 года мы решили возобновить проект. В первую очередь перед нами стоит цель исправить такие недостатки как: морально устаревший программный интерфейс (API), привязка к локально установленному GraphViz и отсутствие использования возможностей GraphViz новых версий.

Применение в преподавании и использование в Java Community

Изначальная идея возрождения проекта возникла в рамках подготовки курса Core Java Ивана Пономарёва для МФТИ. Диаграммы

полученные с помощью LJV применялись на соответствующих лекциях о стандартных структурах данных в Java. Кроме того, была сделана интернет-публикация [4] и сделано выступление на профессиональной Java конференции SnowOne 2021 [5, 6], которые вызвали определённый интерес в сообществе Java-разработчиков.

Принцип работы

В режиме runtime инструмент строит внутреннее представление графа объектов памяти, используя Reflection API. Мы рекурсивно начинаем от передаваемого объекта и заходим в каждый связанный объект и собираем необходимую информацию. По ходу работы мы также формируем GraphViz представление с собранной информацией и сохраняем в текст на языке DOT, чтобы затем на его основе пользователь мог самостоятельно нарисовать граф в установленном локально приложении или с помощью онлайн-сервисов.

Изменения

Нами проделаны следующие задачи:

Мы обновили проект до Java 11, использовав практики построения API, которые выработались за 16 лет, а также новые возможности языка. Мы не стали писать на Java 14, так как данная версия ещё не стала популярнее Java 11, а мы ставим цель охватить больше людей, в частности преподавателей и студентов.

Полностью перестроен принцип генерации графа с использованием новых возможностей GraphViz, которые, в свою очередь, позволили легко добавить некоторую новую функциональность -- например, выбор направления отрисовки графа. Поэтому теперь пользователям доступна ещё больше возможных вариантов графа.

Изменён API инструмента на удобный method chaining, чтобы добавлять новые параметры к одному объекту.

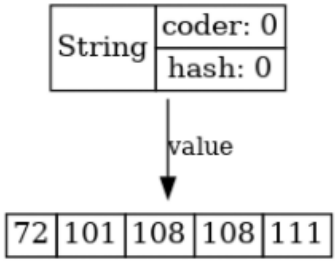
А также код и документация выложены на Github, а сам проект доступен для скачивания в Maven Central.

Как использовать

У объекта LJV есть метод drawGraph куда мы передаём объект, который мы хотим визуализировать. Метод возвращает строку, кото-

рую мы загрузили в онлайн-сервис `dreampuf.github.io` и получили изображение:

```
String graph = new LJV().drawGraph("Hello");
```



Также перед вызовом drawGraph можно передать различные параметры:

```
String graph = new LJV()
  .setTreatAsPrimitive(Integer.class)
  .addFieldAttribute({"next", "color=red,fontcolor=red")
  .addFieldAttribute("prev", "color=blue,fontcolor=blue")
  .addFieldAttribute("first", "color=red,fontcolor=red")
  .addFieldAttribute("last", "color=red,fontcolor=red")
  .drawGraph(linkedList);
```

Оформление библиографии

Если Вам известно, как работать с библиографией в программе OpenOffice.org, всё дальнейшее можно пропустить. Если же нет, то,

скорее всего, у Вас стоят настройки по умолчанию, которые мало подходят для русскоязычных пользователей.

Для этого идём в меню «Сервис — Параметры» и выбираем пункт «База данных OpenOffice.org». Первым пунктом в табличке будет идти база Bibliography, нам понадобится её «Правка ...». Из открывшегося окна копируем путь до файла базы данных и, собственно, больше нам окна «Параметров» не понадобятся, их можно закрыть.

А с помощью скопированного пути открываем базу данных для редактирования через обычный диалог «Открыть ...». Потребуется изменить две вещи. Первая: «Правка — База данных — Свойства — Дополнительные настройки — Кодировка», выбираем UTF-8 или что ближе по духу. Вторая: идём в таблицы, открываем таблицу biblio для редактирования, изменяем максимальную длину поля title с 75 до 254 символов. После этого базу данных можно закрывать.

Для работы с содержимым библиографической базы данных нужно зайти в меню «Сервис — База данных библиографии». Там всё достаточно очевидно, потому просто заполняете записи данными использованной литературы. Вставлять ссылки в тексте на библиографические источники нужно через меню «Вставка — Оглавление и указатели — Элемент списка литературы», вот так: [1]. Данные источника будут внесены в документ и для корректного отображения Вам не потребуется распространять базу данных вместе со своим документом. Для добавления самого списка нужно выбрать «Вставка — Оглавление и указатели — Оглавление и указатели ...», указав тип указателя «Библиография». Потом уже, через правку списка (по правой кнопке мыши, «Правка указателя»), стилей и т.п. можно добиться нужного результата. Он может выглядеть вот так:

Литература

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

[2] Текущая версия проекта, https://github.com/atp-mipt/ljv

[3] Изначальная версия проекта, https://www.cs.auckland.ac.nz/~j-hamer/LJV.html

[4] John Hamer, Visualizing Data Structures as Graphs, ACE 2004 conference, https://www.cs.auckland.ac.nz/~j-hamer/ACE04-paper.pdf

[5] Ivan Ponomarev, LJV: What We Can Learn From Java Data Structures Visualization, https://dzone.com/articles/what-can-we-learn-from-java-data-structures-visual

[6] Иван Пономарёв. LJV: Чему нас может научить визуализация структур данных в Java -- доклад на Java-конференции SnowOne, 2021 г., https://snowone.ru/speakers/ivan_ponomarev

[7] Иван Пономарёв. LJV: Чему нас может научить визуализация структур данных в Java -- слайды к докладу на Java-конференции SnowOne, 2021 г., https://inponomarev.github.io/ljvtalk/

Алексей Федорчук, Ольга Ломова
Москва

Проект: Linux для пенсионеров https://www.cinia.ru/

Базовые дистрибутивы «пенсионерского» Linux'а

Аннотация

Рассмотрены критерии выбора базового дистрибутива «пенсионерского» Linux'а и рабочего окружения для него. В качестве второго нами принята среда Cinnamon, оптимально поддерживаемая в системе Ubuntu Cinnamon Remix. Обоснованию именно такого решения и посвящён предлагаемый доклад.

Пенсионер, впервые приобщающийся к Linux'у, постоянно сталкивается с необходимостью выбора — дистрибутива, рабочей среды, прикладных программ. Конечно, главный выбор, в виде Linux'а, он уже сделал. А далее будет лействовать под влиянием кого-то из заслуживающих доверия «действующих» линуксоидов, имеющих сложившиеся, хотя и обычно субъективные, предпочтения.

Однако далеко не у всех пенсионеров есть такие «доверенные» линуксоиды. Именно для них мы с котом Мануалом и сочиняем этот труд. Конечно, его рекомендации также субъективны. Однако в нём мы попытаемся обосновать, почему наши предпочтения именно такие.

Начнём мы с выбора рабочей среды (десктопа) — вопреки сложившему мнению, он гораздо важнее, нежели выбор дистрибутива. С последним применитель сталкивается в основном при установке системы (которая, скорее всего, будет выполняться тем самым «доверенным» линуксоидом), а также при работе с пакетами. В рабочей же среде применитель проводит всё своё рабочее время.

Выбор дистрибутива следует за выбором рабочей среды ещё и потому, что во многом определяется последней. Ибо, хотя большинство универсальных дистрибутивов поддерживают несколько декстопов, но обычно имеют свою «титульную» рабочую среду.

В настоящее время активно развивается несколько рабочих сред (KDE, GNOME, MATE etc.). Но мы не будем тянуть кота за хвост (тем более что Мануал этого не любит), и сразу скажем: наш выбор — среда Cinnamon. Почему — ответить не трудно.

Во-первых, по функционалу она занимает промежуточное положение между изобилием функций KDE и более простой Xfce. И практика показывает, что функционала её вполне достаточно для большинства применителей (в том числе и пенсионеров).

Во-вторых, по настраиваемости Cinnamon также располагается между KDE (где настроить можно всё) и просто конфигурируемой средой Xfce. Что немаловажно, все настройки в нём выполняются из единого Центра управления (`Параметры системы`), никаких «левых» твиков для этой среды не существует за ненадобностью.

В-третьих, среда Cinnamon за без малого 10 лет своего существования развивалась всегда плавно и предсказуемо — за всю её историю не было ничего подобного «революциям», происходившим в KDE при переходе от версий 3-й ветки к 4-й, или при смене GNOME 2 на GNOME 3.

В-четвёртых, предсказуемость распространяется и на обновление среды: оно привязано в очередному релизу дистрибутива Linux Mint. Каковой, в свою очередь, появляется обычно примерно через месяц после выхода материнской Ubuntu.

Наконец, в-пятых, среда Cinnamon выглядит наиболее привлекательной с эстетической точки зрения — впрочем, это наше с Мануалом субъективное мнение.

Определившись с рабочей средой, остаётся подобрать дистрибутив из числе тех, в которых среда Cinnamon поддерживается должным образом. Число это не велико: по данным Distrowath'a[1], оно укладывается в три десятка. Причём в большинстве систем из этого списка Cinnamon фигурирует в ряду других десктопов по принципу «что-

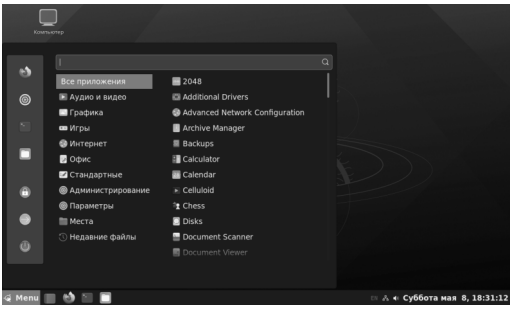
[1] `https://distrowatch.com/search.php?ostype=All&category=All&origin=All&basedon=All&notbasedon=None&desktop=Cinnamon&architecture=All&package=All&rolling=All&isosize=All&netinstall=All&language=All&defaultinit=All&status=Active#simple`

бы было». Дистрибутивов же, где Cinnamon поддерживается хорошо, может по пальцам пересчитать и однорукий.

Первым в этом ряду вспоминается, разумеется, Linux Mint — в рамках этого проекта и зародилась среда Cinnamon, очередные версии которой выходят одновременно с новым релизом материнской системы. Что, во-первых, гарантирует актуальность десктопа, во-вторых, полноту его сборки (в настройках более иных дистрибутивах обычно отсутствую языковые и некоторые административные модули). Кроме того, LM — одна из самых беспроблемных систем как при установке, так и при последующей настройке. Наконец, LM — хорошо документированная и собственными средствами, и сторонними (например, в онлайновой книжке одного из авторов этих строк: Linux Mint и его Cinnamon[2]).

Всё это позволяет рекомендовать LM в качестве основы «пенсионерского» Linux'а, в том числе для совсем начинающих и особо ленивых применителей (что мы с Мануалом обычно и делали). Однако за годы своего развития он оброс большим количеством дистрибутив-специфичных утилит, преимущественно Python-скриптов, что отрицательно сказывается на его простоте и производительности (особенно на маломощных машинах). И наши рекомендации несколько изменились.

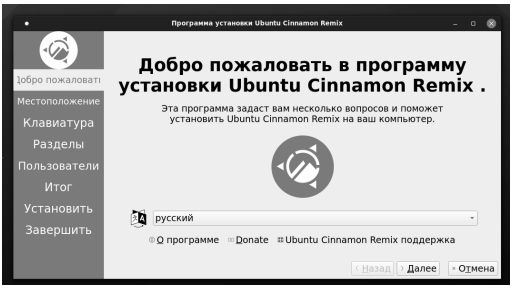
Ибо в последние полтора года у LM появилась альтернатива Ubuntu Cinnamon Remix (далее UCR):



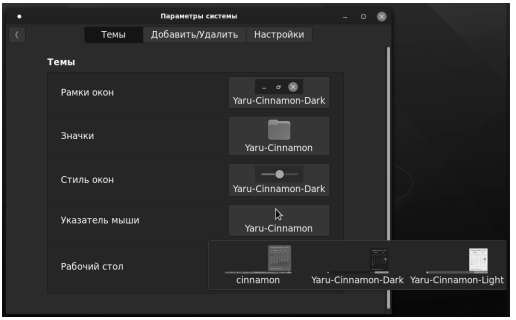
Это очень простая система, слагаемая из базовой Ubuntu (пакеты — из официальных её репозиториев) и полудюжины пакетов из

[2] http://alv.me/?p=8406

собственного PPA-репозитория[3], содержащих Cinnamon и инсталлятор на базе Calamares'a. Последний — единственный дистрибутив-специфический компонент системы:



Если не считать, конечно очень скромного набора тем оформления:



UCR пока не имеет официального статуса клона Ubuntu, однако разработчик рассчитывает таковой получить.

[3] https://launchpad.net/~ubuntucinnamonremix/+archive/ubuntu/all?field.series_filter=hirsute

Андрей Михеев
Москва, RunaWFE Free
http://runawfe.org

Проекты производственной практики и ВКР, связанные с разработкой свободной системы RunaWFE Free

Аннотация

Проект RunaWFE Free в течение нескольких лет привлекает студентов — программистов к разработке системы. Задачи на производственную практику и ВКР ставятся так, чтобы студент получил опыт решения какой-то небольшой проблемы «в целом». То есть, проблема формулируется на языке предметной области, детальное ТЗ на кодирование отсутствует, при этом проблема является реальной, её программное решение (если оно будет успешным) войдёт в один из следующих релизов системы. Во время решения задачи студенту даётся большая степень свободы. Предполагается, что он будет решать задачу «без начальника», только с консультантами. Я надеюсь, что пройдя такую практику, студентам в дальнейшем будет легче самим находить интересные востребованные проблемы и ставить задачи как себе, так и другим разработчикам. СПО помогает «мягко» войти в эту деятельность: Все желающие знакомятся с кодом системы и только те, кто готов взяться за решение задачи, привлекаются в проект.

Проблемы современного обучения

В настоящее время обучение как слушателей на коммерческих курсах, так и студентов во многих ВУЗах от принципа «know why» всё больше приближается к принципу «know how». То есть, задачей обучающегося является — очень хорошо выучить уже существующую, не им придуманную технологию (в большинстве случаев — иностранную) и в дальнейшем применять её к решению соответствующего класса задач.

Получившим такое образование оказывается очень сложно решать задачи (выполнять работы), похожих на которых не было ранее. Которые только что возникли в современном обществе. Поэтому для того, чтобы «тренировать» способность студентов самостоятельно решать проблемы, при прохождении у нас производственной практики,

а также при написание ВКР мы стараемся формулировать задачу не в «программистских» терминах, а в терминах предметной области и даём определённую свободу при решении этой задачи, чтобы студенты получили опыт решения какой-то небольшой проблемы «в целом».

То, что мы разрабатываем свободный продукт, оказывается для этого удобным. Задачи у нас достаточно сложные, не у всех студентов получается их делать. Поэтому студентам, проявившим интерес к проекту, мы сразу даём ссылку на открытый репозиторий проекта[1], ссылку над документацию (пользователя, администратора и разработчика)[2] , описание предметной области[3] и примеры решаемых задач[4]. После знакомства с этими материалами студенты уже имеют некоторое представление об ожидающей их деятельности и принимают решение более ответственно.

Завершённые и текущие «студенческие» проекты

В настоящем докладе предполагается рассказать о следующих «студенческих» проектах:

1. Создание внутреннего хранилища данных
2. Реализация элемента «бизнес-правило»
3. Создание чата участников экземпляра бизнес-процесса
4. Интеграция системы с Искусственным Интеллектом
5. Создание глобальных разделов
6. Создание специального бота для внутреннего хранилища данных
7. Реализация событийного подпроцесса

Первые четыре из этих проектов реализованы, пятый и шестой выполняются, а седьмой ещё только предложен студентам.

Создание внутреннего хранилища данных

Цель работы — создать в системе внутреннее хранилище данных, представляющее собой наборы объектов предметной области. Реализовать в этом хранилище возможность создания, изменения, удаления

[1] `https://github.com/processtech`
[2] `https://runawfe.ru/`документация2
[3] `https://runawfe.ru/TrainingMaterials_4_4_1_Theory`
[4] `https://sourceforge.net/p/runawfe/discussion/426100`

и чтения бизнес-объектов из экземпляров бизнес-процессов, выполняющихся на сервере. Разработать в графическом дизайнере интерфейсы, при помощи которых пользователи смогут удобным образом задавать характеристики бизнес-объектов и операций с ними. Добавить установку внутреннего хранилища в дистрибутив системы.

Эта задача реализована студентом университета ИТМО. Примеры разработанных интерфейсов приведены на Рис. 1.

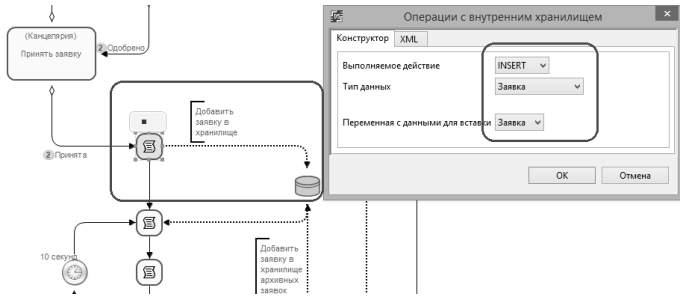


Рис. 1: Примеры интерфейсов, разработанных для решения задачи создания внутреннего хранилища.

Реализация элемента «бизнес-правило»

Цель работы — реализация в системе элемента графической нотации BPMN «Бизнес-правило». Функциональность элемента заключается в исполнении определённой формулы, при выполнении заданного условия.

Эта задача реализована студентом ИАТЭ НИЯУ МИФИ. Примеры разработанных интерфейсов приведены на Рис. 2. и Рис. 3.

Создание чата участников экземпляра бизнес-процесса

Чат участников экземпляра бизнес-процесса позволяет участникам экземпляра бизнес-процесса во время исполнения бизнес-процесса обмениваться друг с другом сообщениями, не изменяя состояние бизнес-процесса (т.е. не перемещая точки управления по его схеме).

Задача реализована студентами НИУ МИЭТ. Примеры разработанных интерфейсов приведены на Рис. 4.

Интеграция системы с Искусственным Интеллектом

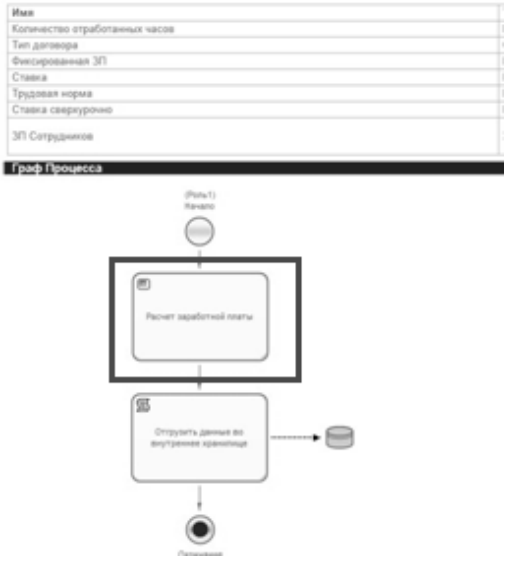


Рис. 2: Пример использования элемента «бизнес-правило» в экземпляре бизнес-процесса.

Было реализовано взаимодействие системы с двумя свободными библиотеками, реализующими Искусственный Интеллект в виде деревьев решений. Так как API к этим библиотекам реализован для Python, а система RunaWFE Free разработана на Java, для взаимодействия был создан Python сервер, соответствующие коннекторы и разработан протокол взаимодействия RunaWFE Free и Python сервера.

Библиотеки были обучены, запросы к ним «вставлены»в бизнес-процессы через соответствующие элементы бизнес-процесса (задачи — сценарии). Данное решение показало результаты, оценённые бизнесом. На основе этого решения планируется организовать стартап.

Задача реализована студентами НИУ ВШЭ. Схема взаимодействия бизнес-процессов системы RunaWFE FREE с Искусственным Интеллектом приведена на Рис. 5.

Создание глобальных разделов

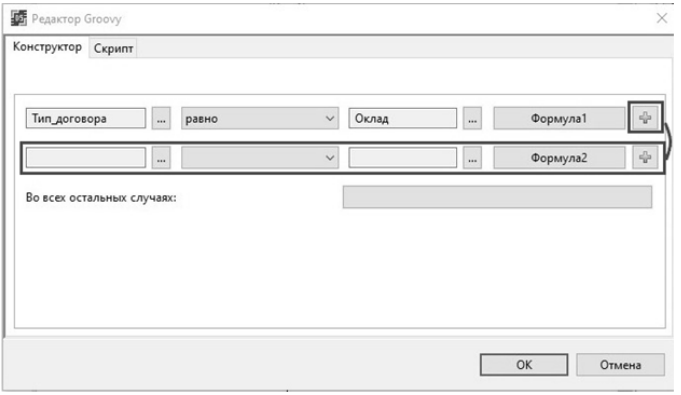


Рис. 3: Конфигурация элемента «бизнес-правило».

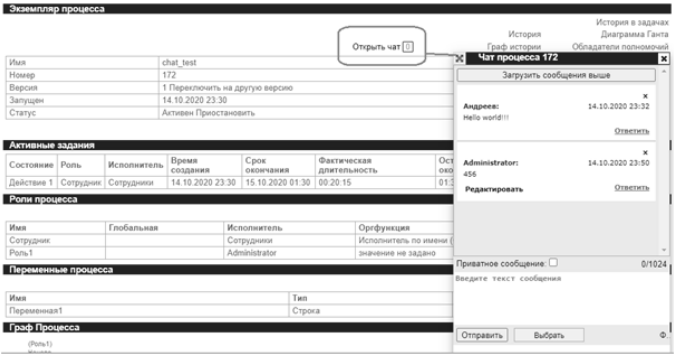


Рис. 4: Чат участников экземпляра бизнес-процесса.

В этой задаче для того, чтобы не заводить во многих бизнес-процессах одни и те же объекты, надо добавить в графический дизайнер глобальные роли, глобальные переменные и глобальные типы переменных. Команда создания глобального раздела приведена на Рис. 6.

В настоящее время задача реализуется в НИУ МИЭТ.

Создание специального бота для внутреннего хранилища данных

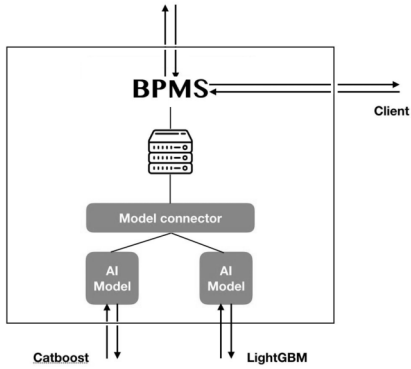


Рис. 5: Интеграция RunaWFE Free с Искусственным Интеллектом.

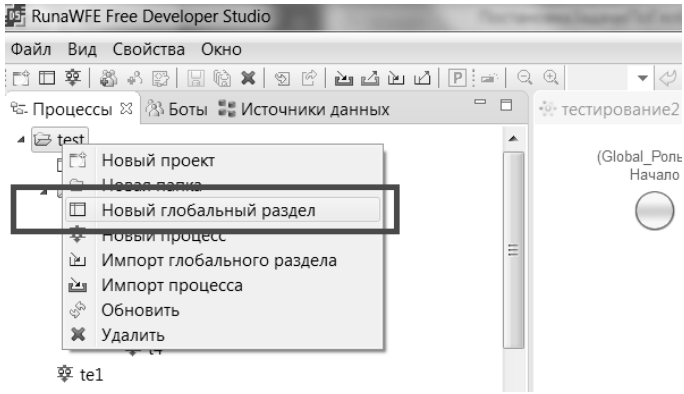


Рис. 6: Команда создания глобального раздела.

Реализованным для внутреннего хранилища задачей-сценарием не всегда можно обойтись. Для использования бизнес-транзакций требуется другая сущность — бот (автоматический исполнитель заданий). Использовать текущего бота для внутреннего хранилища неудобно: имена формальных параметров бота и полей таблиц в конфигурации заданий бота внутреннего хранилища надо вводить «руками» с клавиатуры, т. к. бот разрабатывается независимо от бизнес-процессов и

«не знает» их бизнес-объектов. При этом пользователи иногда ошибаются, что приводит к ошибкам при выполнении бизнес-процессов.

В данной задаче предлагается вводить эти параметры не «руками», а выбирать из списков. Ошибки неправильного ввода с клавиатуры будут в этом случае исключены. Объекты-параметры заданий бота внутреннего хранилища предлагается выбирать из присоединённых к боту глобального раздела.

В настоящее время эта задача реализуется в НИУ МИЭТ.

Реализация событийного подпроцесса

Цель работы — реализация в системе элемента графической нотации BPMN «Событийный подпроцесс» (как в графическом дизайнере, так и на сервере). Этот элемент похож на элемент «внутренний подпроцесс», но граница этого элемента — пунктирная.

У элемента не может быть ни входящих, ни исходящих переходов. Он должен находиться на схеме изолировано от основного графа процесса. Стартовыми элементами событийного подпроцесса могут быть только непрерывающие событийные элементы нотации BPMN. Все остальные элементы в палитре событийного подпроцесса — такие же, как в обычном внутреннем подпроцессе. На схеме событийного подпроцесса всё располагается так же, как в обычном внутреннем подпроцессе.

Событийный подпроцесс «работает» следующим образом:

Если в процессе, на схеме которого расположен событийный подпроцесс, находится хотя бы одна точка управления, то стартовый узел событийного подпроцесса находится в ожидании возникновения события, на которое он настроен. Если все точки управления покинули процесс, в котором находится событийный подпроцесс, то стартовый узел подпроцесса перестаёт быть активным и все наступающие события игнорирует.

Если стартовый узел событийного подпроцесса сгенерировал точку управления (при наступлении соответствующего события), то дальше событийный подпроцесс выполняется так же, как и обычный подпроцесс-композиция, за исключением того, что точка управления, пришедшая в узел-окончание, просто удаляется.

Эта задача пока ещё не назначена никому из студентов.

Преимущества для студентов от выполнения курсовых, производственной практики и ВКР в свободном проекте RunaWFE Free

1. Не нужно сложным образом получать доступ к коду. Познакомиться с кодом студент может ещё до того, как обратится в команду проекта.
2. Работа в проекте удалённая, происходит через интернет, студенту не требуется куда-то ездить и тратить на это время.
3. Студенту легко включить результаты работы в портфолио своих проектов. Репозиторий открытый, любой желающий может посмотреть коммиты студента.
4. Студенты имеют возможность продемонстрировать качество своего кода будущему работодателю при устройстве на работу
5. Мы стараемся составлять задачи для студентов таким образом, чтобы они соответствовали учебной программе ВУЗа, в котором обучается студент. Чтобы эту работу можно было зачесть как курсовые, производственную практику и ВКР. То есть, студенту не нужно искать «дополнительное» время на участие в проекте, работа в проекте войдёт в «учебное» время, которое студенту в любом случае надо будет потратить.
6. Стажировка у нас оплачивается студенту. «Студенческая» ставка небольшая, но даёт некоторые средства к существованию, что для студента может быть важно.
7. Работая в свободном проекте, студент может участвовать в конференциях по использованию СПО в учебном процессе и (если у него получится сделать что-то серьёзное) — в конференциях разработчиков открытого кода, где он сможет показать свои решения и обменяться опытом с серьёзными разработчиками-профессионалами.

Литература

[1] Ссылки на сайты проекта RunaWFE Free: `http://runawfe.org` , `http://runawfe.ru`

[2] Михеев А. Г., Орлов М. В. Система управления бизнес-процессами и административными регламентами. // Программные продукты и системы, № 3, 2011, сс. 126–130.

[3] Выступлении команды студентов НИУ МИЭТ, разрабатывающих «Чат» на XVI конференции разработчиков СПО в г.Калуга: http://institut-spintex.ru/news/education/my-na-xvi-konferentsii-razrabotchikov-spo.html

Алексеев Михаил
Санкт-Петербург, ФСПО НИУ ИТМО, ООО «Процессные технологии»

Проект: RunaWFE Free https://runawfe.org/rus/

Реализация внутреннего хранилища бизнес-объектов в свободной системе управления бизнес-процессами RunaWFE Free

Аннотация

При внедрении RunaWFE на предприятиях было выявлено, что наибольшую сложность составляет организация работы с слоем бизнес-объектов. В докладе описана реализация внутреннего хранилища бизнес-объектов в системе RunaWFE Free.

При внедрении RunaWFE на предприятиях было выявлено, что наибольшую сложность составляет организация работы с слоем бизнес-объектов.

Слой бизнес-объектов — это информация о ресурсах организации, предназначенная для персистентного хранения. Например, данные о банкоматах, товарах и пр.

Жизненный цикл бизнес-объектов выходит за рамки жизненного цикла бизнес-процессов. Таким образом, с одним и тем же бизнес-объектом могут взаимодействовать разные экземпляры бизнес-процессов.

Для решения проблемы в системе RunaWFE Free было решено реализовать внутреннее хранилище бизнес-объектов, удовлетворяющее следующим требованиям:

1. В качестве персистентного хранилища должны использоваться Excel файлы.
2. Внутреннее хранилище должно автоматически разворачиваться при установке RunaWFE Free, не требуя от пользователей дополнительных действий, но быть кастомизируемым под нужды пользователей.

3. Внутреннее хранилище должно быть реализовано с минимальными трудозатратами, используя существующие компоненты RunaWFE.

Хранилище на основе Excel файлов было выбрано в учебных целях, чтобы студенты финансово-бухгалтерских специальностей могли легко проверить результаты работы бизнес-процессов, не обладая специфическими знаниями теории баз данных.

Работу с внутренним хранилищем было решено реализовать при помощи следующих элементов:

1. Обработчик внутреннего хранилища.
2. Иконка data store из BPMN и пунктирная стрелка.
3. Признак хранения во внутреннем хранилище для пользовательских типов данных.

Обработчик может быть сконфигурирован для выполнения 4 операций: insert, select, update, delete.

Выбор подмножества операций должен происходить автоматически при связывании иконки data store с обработчиком внутреннего хранилища с помощью пунктирной стрелки:

```
data store → обработчик = {select}
обработчик → data store = {insert, update, delete}
```

В качестве основы для обработчика внутреннего хранилища был взят существующий обработчик внешнего хранилища, конфигурация которого для операции update представлена на Рис. 1.

Способ конфигурации обработчика внешнего хранилища обладает следующими недостатками:

- пользователю предлагается слишком много полей для ввода, что затрудняет использование обработчика внешнего хранилища новыми пользователями;
- поле «Условие» принимает на ввод строку определённого формата, в котором легко допустить ошибку;
- выбраны не самые удачные названия настроек для секции «Атрибут», которые меняют своё значение в зависимости от выполняемой операции, что ещё сильнее усложняет понимание конфигурации;

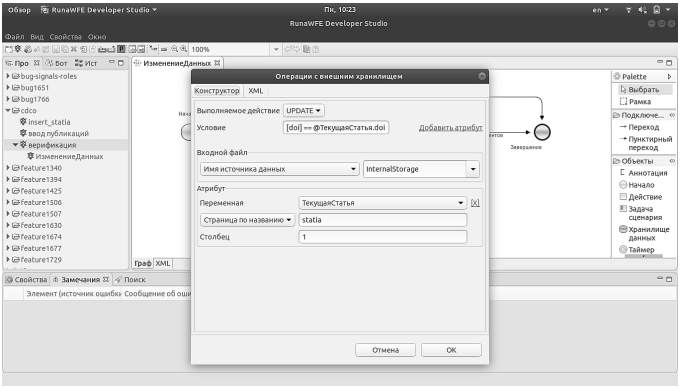


Рис. 1: Конфигурация обработчика внешнего хранилища

Для переиспользования серверного кода обработчика внешнего хранилища было решено сохранить формат конфигурации обработчика внешнего хранилища, улучшив его usability в обработчике внутреннего хранилища.

В ходе анализа конфигурации обработчика внешнего хранилища были определены следующие этапы улучшения usability,:

1. Автоматически заполнять параметр конфигурации «Входной файл» (источник данных) специальным значением *InternalStorage* при связывании обработчика внутреннего хранилища с иконкой data store, таким образом убрав данный параметр из конфигурации.

2. Упразднить секцию «Атрибут», введя параметр «Пользовательский тип данных», который может принимать любое значение из множества пользовательских типов данных, имеющих признак хранения во внутреннем хранилище.

3. Минимизировать количество отображаемых параметров конфигурации, оставив только необходимые для совершения операции. В таблице 1 представлено соответствие необходимых параметров для каждой из операций.

4. Типизировать параметр «Условие» на основании выбранного пользовательского типа данных.

Операция	Предикаты	Переменная с данными/ переменная-результат
insert	-	+
select	+	+
update	+	+
delete	+	-

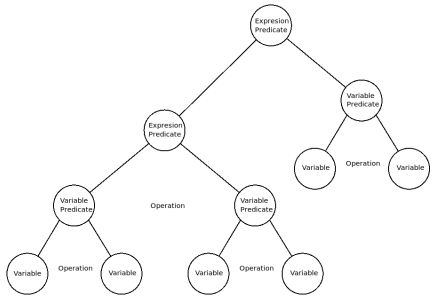


Рис. 2: Дерево предикатов

Параметр «Условие» в конфигурации обработчика внешнего хранилища представляет собой строку вида

```
[атрибут_пользовательского_типа_данных] операция_сравнения @переменная_сравнения
                                                операция_объединяющая_предикаты
[атрибут_пользовательского_типа_данных] операция_сравнения @переменная_сравнения
...
```

Которая может быть описана с помощью бинарного дерева, представленного на Рис. 2.

Логическое представление дерева предикатов позволяет осуществлять его обработку, в том числе типизацию отдельных предикатов.

При реализации описанной функциональности были использованы следующие технологии:

- Java 8;
- Apache Poi 3.17;
- Apache Struts 1.3.8;
- Apache Commons 2.6;

- Spring 3.1.2;
- Lombok 1.18;
- Dom4j 1.6.1;
- Eclipse Tycho 1.5.1.

Результаты

На Рис 3–6 представлены скриншоты элементов системы, использующихся для работы с внутренним хранилищем. Модифицированные конфигурации обработчиков не допускают ввода произвольного текста, дают только возможность выбора из существующих элементов. Структура конфигураций сильно упрощена за счёт использования направленности соединения задачи-сценария с иконкой базы данных пунктирной стрелкой.

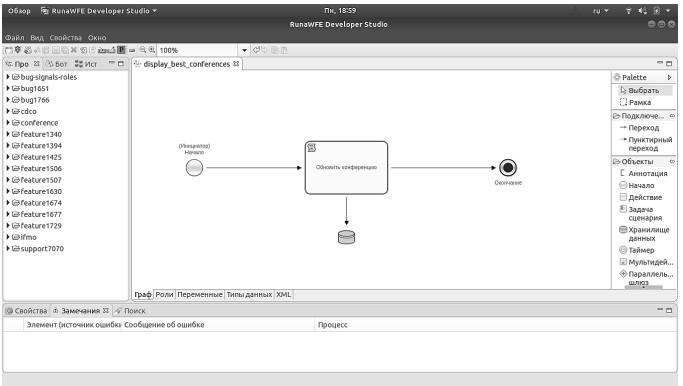


Рис. 3: Схема бизнес-процесса с использованием BPMN data store.

Заключение

В результате реализации внутреннего хранилища пользователям стал доступен удобный способ работы с слоем бизнес-объектов, позволяющий в декларативной форме описывать операции над бизнес-объектами и скрывающий низкоуровневые детали реализации.

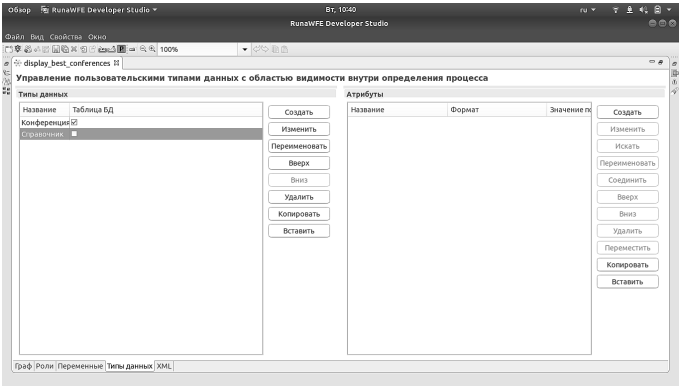


Рис. 4: Признак хранения во внутреннем хранилище.

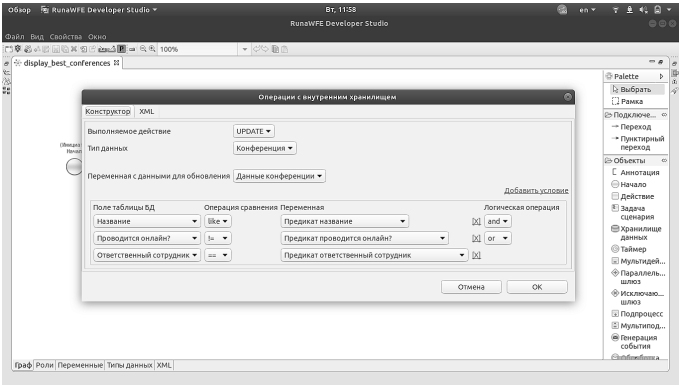


Рис. 5: Конфигурация обработчика внутреннего хранилища.

Литература

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

[2] Ссылка на сайт проекта RunaWFE Free: `http://runawfe.org/rus`

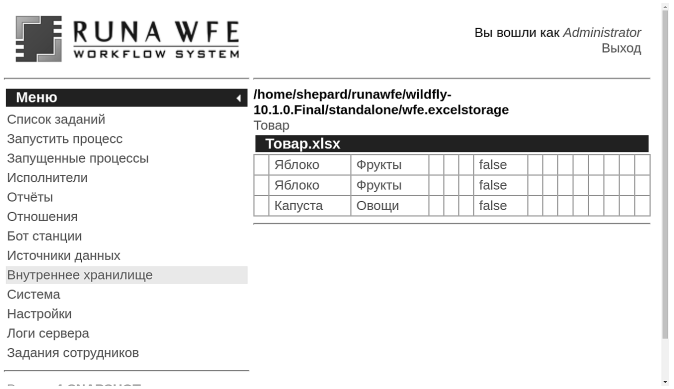


Рис. 6: Содержимое внутреннего хранилища.

Валерий Лаптев, Руслан Шарифулин
Астрахань, Астраханский государственный технический университет

Drakon IDE — среда для обучения алгоритмизации

Аннотация

Описывается разработка среды Drakon IDE, предназначенная для обучения алгоритмизации на графическом языке ДРАКОН. Сформулированы требования к среде, определён минимальный набор икон и разработан интерпретатор дракон-схем типа «Примитив». Разработка осуществляется на языке Dart.

На кафедре АСОИУ Астраханского технического университета возникла потребность в реализации специализированной интегрированной среды для обучения начинающих программистов алгоритмизации. В качестве визуального алгоритмического языка выбран язык ДРАКОН [1]. Этот язык был выбран по нескольким причинам. Во-первых, он близок к блок-схемам, с которыми знакомы почти все студенты, поступившие на ИТ-направления. Во-вторых, Дракон-схемы, в отличие от блок-схем, строятся по строгим правилам. В-третьих, ДРАКОН является российской разработкой.

Требования к интегрированной среде

Первоначально были сформулированы требования к интегрированной среде. Среда должна обеспечивать пользователю следующие минимальные возможности:

- работа с проектами;
- работа со схемами;
- накопление компонентов схем в библиотеке;
- выполнение схемы;
- конвертирование схемы в программу на языке программирования.

Проект может содержать одну или несколько схем. Типичные действия с проектами: создать проект, добавить/удалить схему, сохранить проект, открыть/закрыть проект.

Работа со схемами — это типичные операции создания, редактирования и сохранения схемы. Очевидно, что эти операции должен обеспечивать специализированный графический редактор.

Среда должна обеспечивать накопление библиотеки стандартных алгоритмов и элементов алгоритмов.

Для проверки схемы требуется выполнение сконструированного алгоритма. Выполнение возможно в различных режимах: непрерывное, непрерывное с задержкой, пошаговое.

Конвертирование схемы в программу на языке программирования способствует лучшему пониманию алгоритмов и программ. Кроме того, это позволит получать исполняемый файл для Дракон-схемы.

Интегрированная среда также должна иметь современный, наглядный и информативный пользовательский интерфейс.

Как выяснилось в результате детального анализа, в настоящее время ни одна из существующих систем не обеспечивает выполнение Дракон-схем.

Описание модели схемы

Для работы со схемами разработана модель. Каждой иконе в схеме соответствует объект-икона в модели, которая содержит полную информацию, необходимая для изображения, интерпретации и конвертации иконы. Все иконы, кроме иконы «Вопрос», имеют один вход и один выход. Икона «Вопрос» имеет один вход и два выхода.

Дракон-схемы бывают двух видов: схемы типа «примитив» и схемы типа «силуэт». Модель примитива представляет собой последовательность объектов-икон. Схема типа «силуэт» имеет ветки, которые делят схему на смысловые части. Модель ветки — это последовательность объектов-икон, а весь силуэт представляет собой последовательность веток (последовательность последовательностей икон).

Разработка интерпретатора Дракон-схем

Разработка среды началась с разработки первой версии интерпретатора. Был определён минимальный набор икон для интерпретации схем типа «Примитив»:

- икона «Заголовок»;
- икона «Действие»;
- икона «Вопрос»;
- икона «Полка»;
- икона «Конец».

Для каждой иконы было определены допустимые операции в иконах.

Икона «Заголовок» обозначает начало алгоритма и является хранилищем переменных Дракон-схемы. Количество переменных может быть произвольным. Переменные могут иметь целочисленный, вещественный и логический типы. Вещественные и логические переменные задаются с типом, целые — без типа. Переменным присваивается начальное значение.

Иконы «Действие», «Вопрос» и «Полка» могут содержать только по одному выражению.

Икона «Действие» содержит выражение присваивания переменным Дракон-схемы некоторого значения. Преобразования типов не выполняется.

Икона «Вопрос» служит для сравнения значений двух переменных. Результатом операции сравнения является значение логического типа: истина (true, да, верно) или ложь (false, нет, неверно).

Икона «Полка» содержит выражение с арифметической операцией с двумя переменными. Результат вычисления операции присваивается переменной. Преобразования типов не выполняется.

Рис. 1: Дракон-схема и её текстовое представление

Для отладки первой версии интерпретатора было разработано текстовое представление икон и схемы типа «Примитив». На рисунке 1 показана дракон-схема и её текстовое представление.

Текущее состояние проекта

Первоначальная версия интерпретатора разрабатывалась на языке C++ с использованием библиотеки Qt. Однако в результате более глубокого анализа функционала разрабатываемой среды было принято решение переписать интерпретатор на языке программирования Dart с использованием Flutter SDK. В настоящее время первая версия интерпретатора полностью реализована.

На этом же инструментарии в настоящее время выполняется разработка специализированного графического редактора (см. рис. 1), а также конвертера в учебный язык программирования Slang. Рассматривается разработка конвертеров в промышленные языки программирования, в частности, в C++ и Оберон.

После завершения разработки первой версии среды исходный код будет размещён на github.

Игорь Воронин, Данила Воронин
Шатура, Московской области, Елабуга, РТ, ИПЛИТ РАН, ОЭЗ «Алабуга»
https://www.umkikit.ru/index.php?route=product/category\&path=67

Обучение программированию умных вещей с использованием конечных автоматов. Проект УМКИ

Аннотация

Обсуждаются способы программирования устройств при помощи модели управления в форме конечного автомата. Рассмотрены необходимые ресурсы для создания программ автоматного программирования. Предлагается решение в виде комплекта пособий по образовательной робототехники, как пакет в школьном дистрибутиве UMKI.

Умные вещи — smart shings занимают всё больше наше пространство, автоматизируя рутинные операции. Они включают свет, обогреватель в доме, пылесос или телевизор с мобильного устройства. Так же уже ни кого не удивляет умная теплица возле дома, где выращиваются помидоры, огурцы или перцы на гидропонике. Для управления этим устройствами можно использовать голосовых помощников или приложения на смартфоне.

Для того чтобы всё это легко и просто могло управляться необходимо дать навыки или дать возможность само обучится большому числу пользователей весьма далёких от профессиональных навыков разработки приложений и программирования. Использование автоматного программирования, или конечных автоматов наиболее точно подходит для этих целей.

Термин «автоматное программирование» (АП) введён в 90-х годах прошлого века. Множество споров вокруг него порождены отсутствием его определения, основанного, что было бы логично, на понятиях теории программ. В результате база АП формируется интуитивно, что порождает модель управления вычислениями мало похожую на первоисточник — модель конечного автомата (КА).

Можно привести множество аргументов в пользу перспективности автоматного программирования. Кроме SWITCH-технологии, в качестве «автоматных сред», реализующих автоматный подход в программировании, можно привести пакет Stateflow системы MATLAB или язык UML. В определённой степени альтернативой им может служить среда автоматного Визуального Компонентного Программиро-

вания — ВКПа. Проблема АП лежит и в плоскости создания соответствующего ему языка программирования. Часто его роль исполняют существующие языки, например C++.

В любом языке программирования выделяют:

1. данные,
2. операции
3. управление.

Схематология определяет модель программы в форме схемы программы, которую можно представить как тройку S = (M, A, C), где M — конечное множество элементов памяти, называемых переменными, A — конечное множество операторов, C — управление. Под элементами памяти будем понимать также более сложные объекты, а к операторам будем относить программные функции.

Поставим множеству каналов автоматного управления C операторы из A: входным каналам — *предикаты*, т.е. функции, выполняющие [только] анализ элементов множества M и возвращающие булевское значение, выходным каналам — *действия,* представленные функциями, выполняющими преобразования элементов множества M.

Введённое определение автоматной программы отличает её от других моделью управления, представленного аналитической, графической, матричной и любой другой формой задания автоматов (о способах задания конечных автоматов см.).

При реализации автоматного управления очень удобно описать сразу *все* его свойства, в форме графа, подобно SWITCH-технологии, и далее представить его в табличной форме, подобно таблицам решений, или некая аналитическая форма описания автомата.

Если, допустим, нам необходимо запрограммировать работу «УМНОЙ РОЗЕТКИ УМКИ», которая умеет включать в разное время по расписанию разные устройства от 220 В, или от 12В и производить замеры с датчиков, то как правило простейшие операции программирования — включения реле, или снятие показаний с датчиков — можно реализовать в среде ARDUINO скачивая уже готовые скечи с GITHUB-а. Но когда нам уже нужно объединить весь функционал исполнительных элементов в единый код, то оказывается, что необходимо в цикле реализовать почти систему реального времени. Что для пользователей без подготовки достаточно проблематичная задача.

Таким образом, если подходить к решению поставленной задачи по программированию умных устройств с использованием конечных

автоматов — АП, то задача становится вполне легко реализуемой для любого участника процесса, кто знаком хотя бы на самом базовом уровне с программированием в среде ARDUINO. Для этого ему только достаточно построить схему связей конечных автоматов и используемых устройств, далее она уже достаточно тривиальным образом будет переведена в набор исполняемого кода на основе конструкций CASE-SWITCH

И тогда текущее состояние автомата может быть объектом для организации синхронизации процессов, отражать ход вычислений, использоваться при отладке и т.д. и т.п. Оно устанавливается автоматически в процессе функционирования модели, что уменьшает число операторов программы, упрощает алгоритм и повышает в конечном итоге надёжность программирования.

В нашем случае множество объектов модели определяется числом компонентов, из которых состоит наша умная розетка. Для её программирования очень удобно использовать модель в форме конечных автоматов.

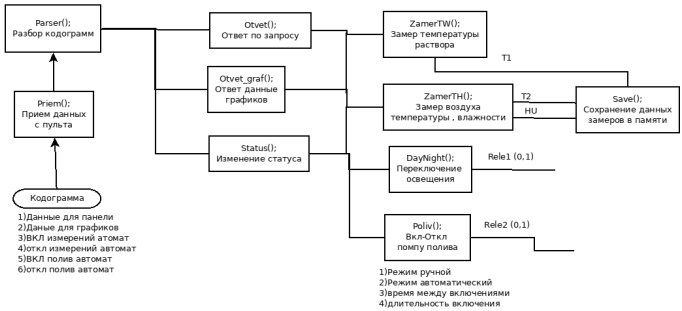


Рис. 1: Схема автоматов управления умной розеткой

Игорь Воронин, Данила Воронин
Шатура, Московской области, Елабуга, РТ, ИПЛИТ РАН, ОЭЗ «Алабуга»
http://astera.laser.ru/umkiurban/ai_out1/

Настройка и использование нейросетей в АЛЬТ для анализа и обработки результатов экспериментов

Аннотация

Системы поддержки принятия решений всё больше входят в нашу повседневность. Мы уже не замечаем, что поисковые системы подсказывают результаты запросов — с учётом наших предпочтений. В нашей повседневности и в работе мы всё больше и больше принимаем решения опираясь на подсказки которые получаем от искусственного интеллект.

Под работой искусственного интеллекта в последнее время подразумевается функционирование нейросети. Поисковики формирую новостную ленту по нашим интересам, основываясь на нейросетях. Алиса из Яндекса больше знает о наших предпочтениях, чем мы сами, поскольку является нейросетью. Нейросеть — это математическая модель, которая получает на вход один набор данных, а на выходе, после ряда преобразований, выдаёт другой набор данных. Такие преобразования осуществляются на основе полученных данных, по стандартным математическим алгоритмам, с учётом так называемых весовых коэффициентов. Иначе говоря, нейросеть — это программа, которая состоит из конечных автоматов как узлов и все эти узлы связаны между собой связями. Каждая связь имеет весовой коэффициент. Узлы объединяются в слои. Чем больше слоёв, тем более точный результат может выдавать такая нейросеть, после обучения.

Прогресс экспериментальных методов привёл к новым знаниям. Однако интерпретация этих экспериментальных знаний, совершение новых открытий, невозможно без дополнительной обработки при помощи программного моделирования. Например, не возможно понять структуру нового материала на атомном уровне без взаимодействия экспериментов и теории. Несмотря на современные методы исследований, которые предлагает сканирующие туннельные микроскопы, просвечивающая электронная микроскопия и др. необходимость использования расчётов теоретических исследований так же возрастает.

Для предсказания заданных параметров нужно провести большие объёмы вычислений и как новые принципы организации расчётных

алгоритмов, для решения этой задачи, используется обучение машинному программированию таких нейросетей (англ. MLP)

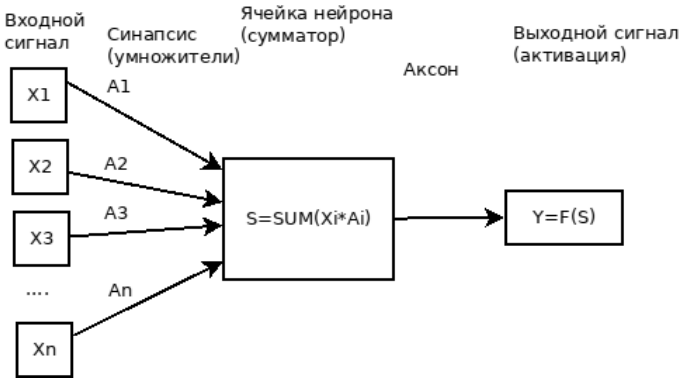


Рис. 1: Структура элементарной ячейки нейросети

Процесс обучения — это выставление весов на связях между узлами используемой нейросети. Для проведения обучения нам нужны две выборки из набора данных (dataset): одна обучающая на которой происходит процесс обучения самой нейросети, вторая тестовая, запустив которую, мы должны убедится, что сеть обучена и, например, распознаёт заданные предметы на проставленных изображениях с достаточной нам точностью. Желательно конечно чтобы точность была 100%, но на всех снимках этого достигнуть практически не возможно, поэтому при задании на разработку нейросети мы должны полагаться на реальные значения. Обычная практика — это 80% снимков на обучение и 20 % на тестирование.

Поскольку количество весовых параметров Xi и Ai часто составляет порядки нескольких тысяч, то зачастую невозможно найти глобальный минимум в этом процессе оптимизации.

Однако найти множество приемлемых локальных минимумов несложно — это будем называть обучением. Обучение нейросети очень требовательно и может занять несколько недель, особенно при использовании больших наборов данных. Поэтому чем более производительные процессоры, чем больше вычислительной мощности у вас

имеется в распоряжении, тем более быстрый процесс будет подготовки сети для работы.

Для обучения и тренировки сети рассчитываются количества весовых параметров, которые заданы архитектурой сети. Именно от количества скрытых слоёв и нейронов на слой зависит производительность всей системы и точность результата

Вариант (а), недообученная сеть, здесь ошибка подгонки велика, и нейросеть не может точно воспроизвести ожидаемый результат.

Вариант (б). Нормально обученная и оттестированная сеть

Вариант (в). Если будет использовано ещё больше параметров, путём увеличения размера нейросети, то может произойти переобучение. Что повлечёт плохое общее представление результата, и, к сожалению, не может быть обнаружено ошибками на этапе тестирования сети.

Существует несколько вариантов обнаружения и предотвращения такого переобучения.

Инструментом для обучения и использования нейросетей, является оболочка Jupyter — это среда для интерактивных вычислений, которая делает использование Python намного проще и интуитивно понятнее. Особенно полезной является его среда записной книжки, которая предоставляет удобный способ экспериментировать с кодом, просматривать результаты и фиксировать на будущее свои успехи.

Обязательные и необходимые компоненты, которые должны быть установлены на сервере для работы с Jupiter: GCC, gfortran, Python, библиотеки линейной алгебры.

```
$ apt-get install gcc gfortran python python-pip libblas-dev liblapack-dev
```

Предпочтения `$ pip install numpy ase f90wrap`

Для оболочек Python (quippy) минимальные требования: Python 3 [NumPy] `http://www.numpy.org` (numpy $\geq$ 1.5.0)

```
$ pip install jupyter
$ jupyter notebook
$ jupyter notebook QUIP/src/GAP/doc/ExamplesIntroduction.ipynb
```

Docker — удобный инструмент для развёртывания вашей обученной нейросети на множестве серверов

Для обучения по образам мы берём фотографии нужных нам объектов. Например мы хотим распознавать маски на лицах. Для обучения сети нам нужно порядка 1000 снимков. Каждый снимок перед

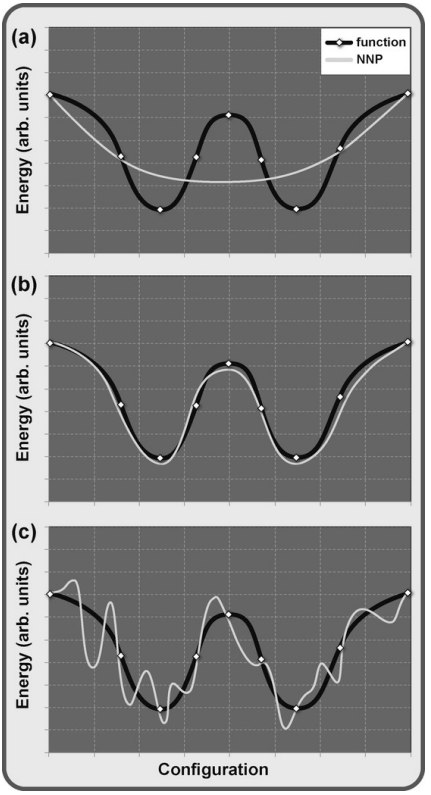


Рис. 2: (а) Недообученная сеть, (б) нормально обученная сеть, (с) Переобученная сеть

началом запуска обучения нам нужно растегеть. Т.е в специальной программе указать теги — координаты на снимке тех мест где точно изображены наши искомые маски на лицах. Это делает оператор вручную. В результате по каждому снимку получаются два файла: один сам с изображением и второй в формате XML с указанием координат на снимке искомого объекта.

После того, как набор снимков мы подготовили должным образом, мы приступаем к запуску на сервере нейросети.

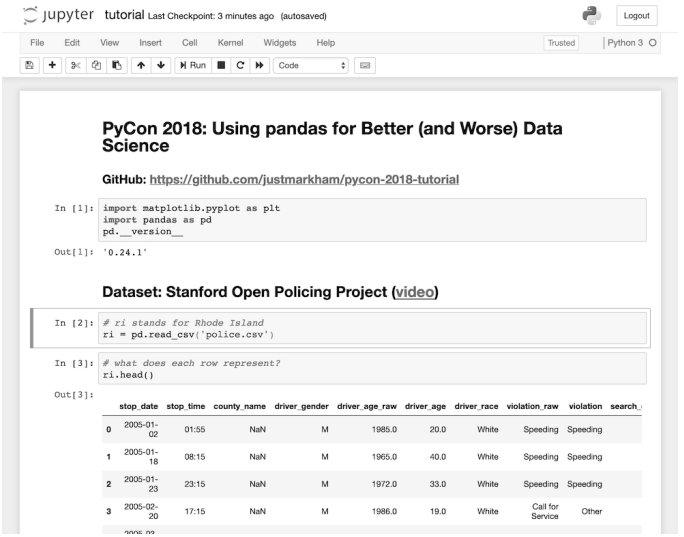


Рис. 3: Окно интерфейса Jupiter

Чтобы использовать готовые библиотеки и не писать программный код с нуля самостоятельно, мы можем выбрать один из двух вариантов. Компания Гугл предоставляет возможность использовать её сервера — с уже предуставновленным программным обеспечением для работы с нейросетями. В этом случае наборы файлов с картинками вы можете положить на свой раздел Гугл диска — и при настройки сети указать логин и путь к этим файлам. Тогда вам ничего никуда больше копировать не придётся, Гугл сам найдёт на своих серверах нужные ресурсы, и подключит их в работу. Но в этом случае могут возникать независимые от вас ограничения в использовании процессорного времени на серверах, поэтому мы предпочитаем использовать собственные сервера, в частности, на российской платформе Alt Liux p9. В этой ОС мы легко настраиваем окружение необходимых библиотек самостоятельно. Alt Liux p9 нам даёт возможность не быть привязанным к условиям компании Гугл, которые при большой загрузке могут начать снижать производительность работы ваших приложений. На своих серверах вы полностью может распоряжаться вашими ресурсами как вам будет нужно.

Для начал работ по развёртыванию сети необходимо подготовиться и ответить на несколько вопросов:

- Какой производительности сервер необходим?
- Какой язык планируется использоваться для работы с нейросетью?
- Какие библиотеки понадобятся для работы?

Основные вычисления производятся, как правило, на CPU сервера, но бывают и такие расчёты, которые можно осуществлять на видео картах — GPU.

Если предполагается осуществлять расчёты на CPU, то обязательно в нём должна быть инструкция AVX, наличие которой мы легко обнаружим следующей командой

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

Для работы нейросети как правило используют язык высокого уровня Python, поскольку очень много различных разработок уже реализованы на этом языке. И достаточно много библиотек имеется в свободном доступе на GitHub.com. Библиотеки от Гугл, которые могут быть использованы для этих целей это: `Numpy` и `Tensorflow`

Для использования различного окружения можно использовать среду `Conda` — это инструмент для управления пакетами и установщик с куда большим функционалом, чем в pip. Conda может обрабатывать зависимости библиотек вне пакетов Python, а также сами пакеты Python.

Рис. 4: Демонстрация работы нейрости по распознаванию образов на снимках. Распознаны: Лев, Лиса, Человек.

Рис. 5: Слева — распознанные изображенния, справа исходные.

Анна Забровская, Владимир Кузьмин, Иван Хахаев
Санкт-Петербург, СПб ГАВМ, СПбГЭТУ «ЛЭТИ»

Применение QGIS и GNU R в исследованиях распространения микроорганизмов, устойчивых к антимикробным препаратам

Аннотация

Рассматривается применение свободных пакетов программ QGIS и GNU R для анализа и визуализации данных по распространению инфекций животных. Показано, что такие пакеты могут успешно применяться в научных исследованиях и в учебном процессе в области ветеринарии и эпизоотологии.

В настоящее время распространение штаммов микроорганизмов, устойчивых (резистентных) к антимикробным препаратам (АМП) носит глобальный характер, и надзор за резистентностью становится всё более актуальным как для медицины в целом, так и для ветеринарии в частности.

Была сформирована база данных [1], включающая результаты изучения чувствительности к АМП 207 штаммов Salmonella, выделенных на территории Ленинградской области с 2004 по 2016 гг. от больных, павших и вынужденно убитых продуктивных животных, из продукции животноводства и кормов. Для вариантов вирусов Salmonella был введён обобщённый критерий устойчивости (резистентности) к набору из 8 групп антимикробных препаратов (АМП): чувствительные (группа 1), устойчивые к 1–2 группам АМП (группа 2), полирезистентные (группа 3), в том числе экстремально резистентные (чувствительные не более чем к 2-м группам АМП, группа 4).

Проводился анализ количества случаев выявления штаммов сальмонелл с различной устойчивостью к АМП, отдельно для каждого года анализируемого периода по районам Ленинградской области.

Для статистической обработки данных был использован пакет GNU R [3] с библиотеками ggplot2 [4], plotrix [5] и riverplot. При использовании базовых возможностей QGIS и открытых данных по административно-территориальному делению Российской Федерации в формате ESRI shape из проекта OpenStreetMap (OSM) на карту Ленинградской области, разделённой на районы, были нанесены данные по случаям выявления штаммов сальмонелл с различной устойчивостью к АМП, отдельно для каждого года анализируемого периода. Для каждого гола формировался отдельный слой. В стилях слоя различными цветами обозначались категории штаммов: чувствительные, резистентные к 1–2 группам АМП, полирезистентные, экстремально резистентные. Отдельным цветом обозначались районы области, в которых сальмонеллы в текущем году не были обнаружены [2]. Для каждого района указывался штамм с наибольшей резистентностью и источник выделения штамма на основе атрибутов записи в данных

Рис. 1: Данные наблюдений за 2009 год

Рис. 2: Данные наблюдений за 2015 год

слоя QGIS. Для публикации полученных карт использовались возможности создания макетов карт. Примеры карт показаны на рис. 1 и 2.

С помощью библиотеки ggplot2 была построена корреляционная матрица для групп штаммов по резистентности к АМП, в качестве

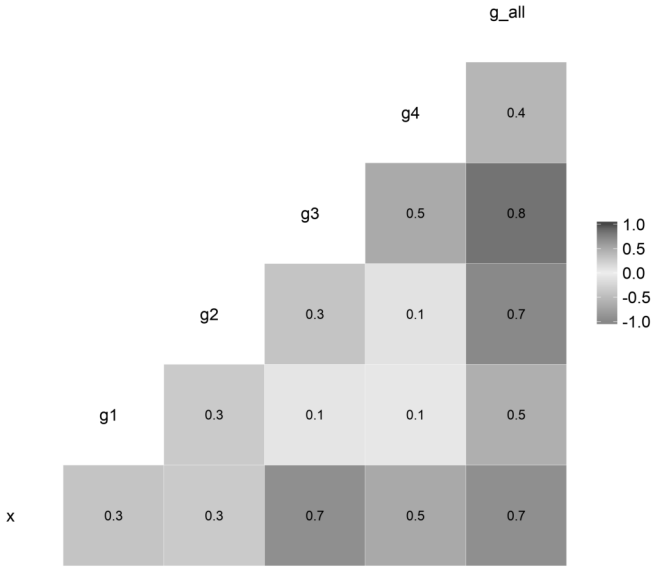


Рис. 3: Корреляционная матрица наблюдений по группам резистентности штаммов

независимой переменной x используются годы наблюдений (рис. 3). В соответствии со шкалой Чеддока [6] корреляцию между общими количеством штаммов (g_all на рис. 3) и годом наблюдения можно оценить как среднюю, а корреляцию между общими количеством штаммов и количеством полирезистентных штаммов можно оценить как высокую. Также имеется корреляция между количеством полирезистентных штаммов и годом наблюдения, которую можно оценить как среднюю. Библиотека plotrix обеспечивает расширенные варианты визуализации результатов статистической обработки данных в R. На рис. 4 и 5 показаны варианты диаграмм, полученных с помощью функции pyramid.plot().

Применение библиотеки riverplot позволило показать, как перетекает максимальный уровень резистентности основных вариантов штаммов от одного источника обнаружения (выделения) к другому по годам наблюдений (так называемая Sankey diagram [7], рис. 6).

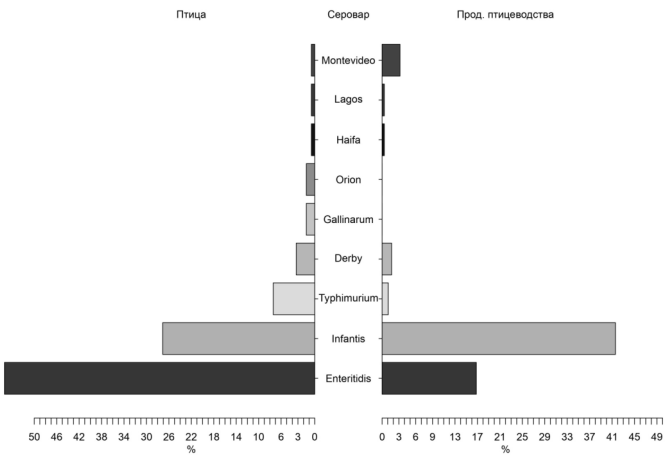


Рис. 4: Штаммы Salmonella птицы и продукции птицеводства

Таким образом, возможности свободных программ QGIS и GNU R в полной мере обеспечивают потребности визуализации данных в эпизоотологии и могут быть использованы в мониторинге ситуации, научных исследованиях и при подготовке специалистов.

Литература

[1] Свидетельство о государственной регистрации базы данных № 2020620404 Российская Федерация. База данных "Salmonella, выделенные от животных, из продукции животноводства и кормов": № 2020620231 : заявл. 20.02.2020 : опубл. 03.03.2020 / *А. В. Забровская, Л. А. Кафтырева, В. А. Кузьмин [и др.].*

[2] *А. В. Забровская, И. А. Хахаев, В. А. Кузьмин, Л. А. Кафтырева.* Пространственная визуализация данных по выделению и чувствительности к антмикробным препаратам штаммов сальмонелл. Вопросы нормативно-правового регулирования в ветеринарии. – 2018. – № 1. – С. 43-45.

[3] R Core Team (2019). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. `https://www.R-project.org/`

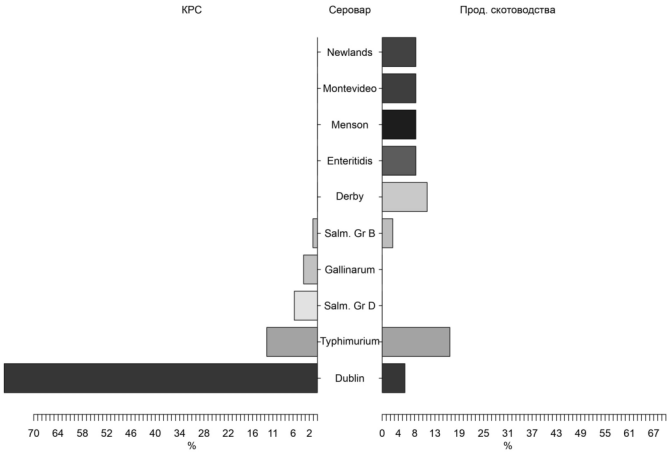


Рис. 5: Штаммы Salmonella продукции животноводства

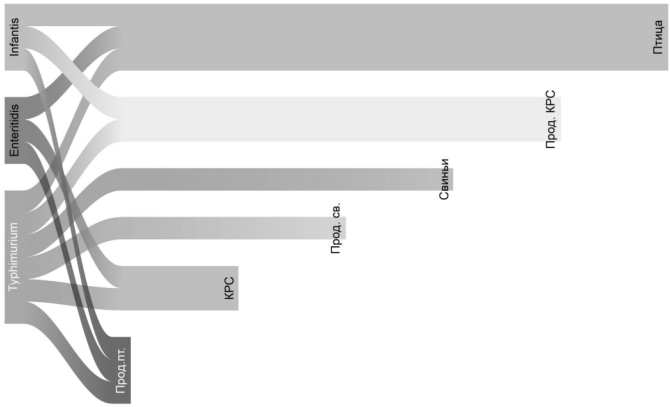


Рис. 6: Распределение основных штаммов по источникам выделения

[4] *H. Wickham.* ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York, 2016.

[5] *J L (2006).* "Plotrix: a package in the red light district of R."R-News, 6(4), 8-12.

[6] *Елисеева, И. И.* Эконометрика: учебник / И. И. Елисеева [и др.]; под ред.

И. И. Елисеевой. 2-е изд., перераб. и доп. М.: Финансы и статистика, 2007. – 576 с.

[7] Sankey diagram. https://en.wikipedia.org/wiki/Sankey_diagram

Кирилл Чувилин
Москва, ООО «Открытая мобильная платформа»
Проект: ОС «Аврора» www.auroraos.ru

Обучение разработке на Qt для мобильных устройств. Что нужно студентам и профессионалам

Qt — это мультиплатформенный фреймворк, предоставляющий широкий набор библиотек и инструментов разработки ПО [1]. Приложения на Qt используются в различных областях: персональные компьютеры [2], мобильные устройства [3], медицинское оборудование [4], автомобильные системы [5]. Сообщество разработчиков насчитывает более полутора миллиона человек, а крупные компании (LG, Panasonic, Mersedes Benz и др.) используют ПО на Qt не только в IT, но и в промышленности. Примеры ПО, написанного на Qt: KDE, Skype, Telegram, Virtual Box, Google Earth и др. Многие дистрибутивы Linux используют Qt как основной фреймворк, в том числе Ubuntu, Fedora, Red Hat и отечественные Заря, ROSA, ALT Linux. Не исключением является и ОС Аврора, разрабатываемая компанией «Открытая мобильная платформа».

Если говорить о мобильных операционных системах, то Qt является основным фреймворком для разработки программного обеспечения для ОС Аврора и Sailfish OS, в отличие от Android и iOS. Учитывая, что ОС Аврора предназначена для использования корпоративными заказчиками, к качеству прикладного ПО для неё предъявляются повышенных требования. Поэтому для компании «Открытая мобильная платформа» особенно важно формирование сообщества квалифицированных разработчиков, которые смогли бы реализовывать сложные индустриальные решения, используя Qt.

С 2016 года сотрудники компании «Открытая мобильная платформа» организуют и проводят учебные мероприятия по разработке, в том числе курсы в вузах и корпоративные тренинги для профессиональных разработчиков. Подготовлены как обучающие материалы

(слайды и тексты лекций, примеры приложений, практические задания), так и варианты тестирования, позволяющие проводить сертификацию разработчиков и преподавателей.

Важно отметить, что технологии, которые используются при разработке на Qt, довольно обширны и включают как различные подходы к шаблонам проектирования (предлагая концепцию сигналов-слотов, а также механизм делегатов для работы с представлениями), так и API для взаимодействия с различными подсистемами и периферией. Последнее особенно важно учитывать при мобильной разработке, когда зачастую не только удобно, но и необходимо использовать возможности устройств: датчики, определение местоположения, технологии связи.

При этом оказывается очень интересно сравнить цели, которые ставят перед собой профессиональные разработчики и профильные вузы. Это формирует и требования к составу учебных материалов. Например, вузы заинтересованы в разносторонней подготовке студентов и формирования у них навыков разработки под различные ОС (с чем Qt неплохо справляется). Профессиональные разработчики, напротив, в первую очередь сосредотачиваются на особенностях целевой операционной системы.

Доклад посвящён как техническим особенностям Qt, которые позволяют на практике освоить важные принципы разработки ПО, так и особенностям учебных материалов, связанным целевой аудиторией, доступными SDK, возможностями кроссплатформенной разработки.

Литература

[1] Qt Product: `https://www.qt.io/product`

[2] Customer stories, desktop: `https://resources.qt.io/customer-stories-desktop-applications`

[3] Customer stories, mobile: `https://resources.qt.io/customer-stories-mobile-apps`

[4] Customer stories, medical: `https://resources.qt.io/customer-stories-medical`

[5] Customer stories, automotive: `https://resources.qt.io/customer-stories-automotive`

Елена Шамаева, Георгий Курячий
Москва, ВМК МГУ
https://gist.github.com/Derinhelm/abd10f66ae505caded35861423959945

MROC3 — не магия, а справедливое слияние очередей

Аннотация

В статье описывается применение в python алгоритма MROC3. С его помощью для иерархии классов однозначным образом определяется очерёдность, в которой среди классов–предков происходит поиск поля. Такой упорядоченный линейный список классов–предков должен удовлетворять интуитивным ожиданиям программиста. В статье алгоритм сводится к справедливому слиянию очередей классов–предков, построенных для родительских классов. Разобраны принципы работы алгоритма, предложен новый метод визуализации объединения очередей.

А люди всё роптали и роптали,
А люди справедливости хотят
В. С. Высоцкий

Появление в python множественного наследования усложнило алгоритм поиска поля, не определённого в классе–наследнике. Поскольку это поле могло быть определено в нескольких родительских классах (см. Рис. 1), появилась необходимость тем или иным способом определять очерёдность просмотра классов–предков.

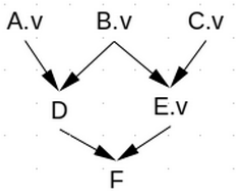


Рис. 1: Иерархия классов.

Порядок просмотра классов удобно задавать с помощью линейного списка[1]: если удастся поместить в *линейный* список все классы из иерархии наследования, то при поиске поля достаточно будет последовательно просматривать эту очередь классов и выбирать *первый попавшийся* класс, содержащий данное поле. В python такой список формируется при создании класса и используется при обращении к любому из полей.

Начиная с версии 2.3, в python для решения этой проблемы используется алгоритм MROC3. Однако в статьях, описывающих этот алгоритм (например, [3] и [4]) практически не объяснено, *почему* в результате данных действий с головами и хвостами различных списков получается искомый список классов, определяющий очерёдность их просмотра при поиске поля.

Список классов, построенный с помощью MROC3, удовлетворяет следующим интуитивным ожиданиям программиста:

- Ожидание №0 — «*однозначность*»: порядок просмотра классов должен определяться однозначным образом, заданным иерархией классов;
- Ожидание №1 — «*дети раньше родителей*»: поиск поля сначала ведётся в дочерних классах, потом — в родительских;
- Ожидание №2 — «*родители в порядке объявления при наследовании*»: приоритетнее классы, стоящие левее при объявлении наследования.

Ожидание №1 задаёт на множестве классов *частичный порядок*: для двух классов, связанных отношением *родители–дети*, можно определить очерёдность просмотра при поиске поля. А Ожидание №0 устанавливает между классами *отношение порядка*, поскольку в однозначно построенном линейном списке для *каждой* пары классов можно установить очерёдность.

В алгоритме MROC3 линейный список классов строится индуктивно. Для класса, созданного без использования наследования, список состоит только из самого класса. При построении линейного списка для класса–наследника используются очереди, ранее построенные

[1] Построение линейного списка вершин графа, удовлетворяющего некоторым условиям (линеаризация), — достаточно известная задача. Например, для Байесовских сетей, перед упрощением формул вероятности строится линейный список вершин графа событий [1]. При топологической сортировке графа также необходимо задать линейный порядок на его вершинах [2].

для его родительских классов. Сначала эти родительские очереди объединяются в одну, затем в начало объединённой очереди добавляется сам класс–наследник. При слиянии родительских очередей должен быть сохранён относительный порядок классов в каждой очереди, иначе для соответствующего родительского класса могут быть нарушены ожидания программиста.

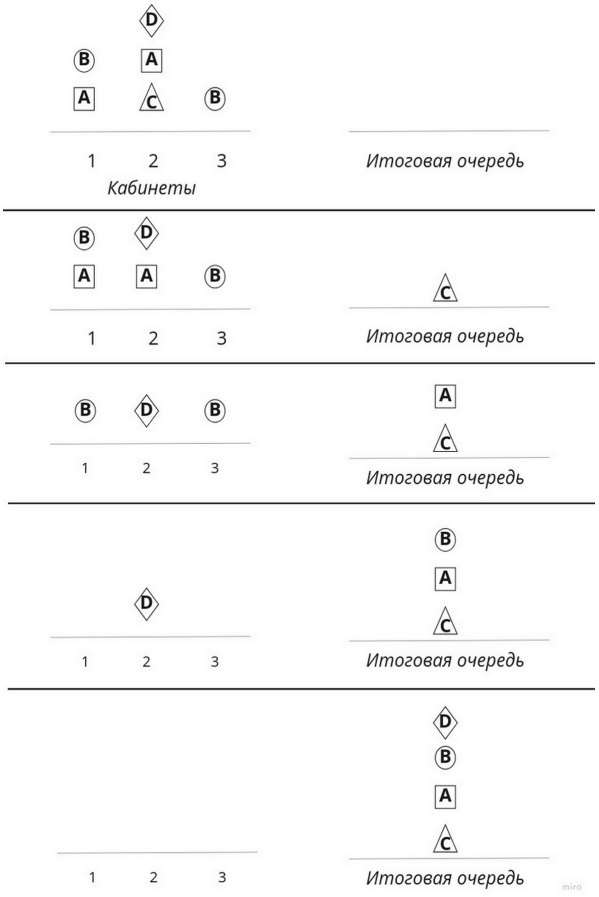


Рис. 2: Слияние очередей.

Слияние родительских очередей аналогично следующей ситуации. Допустим, в поликлинике необходимо объединить несколько очередей (причём каждый пациент ранее мог «занять место» сразу в нескольких очередях). Необходимо сделать так, чтобы *никому не было обидно* и никто не мог возмутиться: «Человек, ранее стоявший за мной, влез передо мной в итоговую очередь!» Пациент может быть перемещён в итоговую очередь только, если во всех очередях, в которых он стоит, перед ним нет других людей. Если несколько человек стоят *первыми во всех своих очередях*, приоритет — у людей из очереди в первый кабинет, потом — во второй кабинет и т.д. (см. Рис. 2). Такое справедливое слияние очередей сохраняет внутреннюю упорядоченность каждой очереди.

Рис. 3: A) Блокировка. B) Противоречие очередей.

Если на каком–то этапе слияния люди блокируют друг друга (см. Рис. 3A), необходимо обратиться к заведующему поликлиникой[2].

Для соблюдения Ожидания №2 создаётся дополнительная очередь из родительских классов в порядке их объявления при наследовании. За счёт справедливости слияния гарантируется соблюдение внутреннего порядка этой очереди (и, следовательно, порядка родительских классов при объявлении наследования)[3]. Эта дополнительная очередь

[2]Для такой иерархии классов python генерирует TypeError

[3]В примере выше дополнительная очередь — это упорядоченная по приоритету кабинетов очередь из тех, кто стоял первый, когда врачи ушли (эти люди

также помогает обнаружить ситуацию, в которой Ожидания №1 и №2 противоречат друг другу. Например, для иерархии *class A: class B(A): class C(A, B):* (см. Рис. 3B).

На схеме ниже (см. Рис. 4) показано, какие аспекты алгоритма MROC3 гарантируют соблюдение ожиданий программиста.

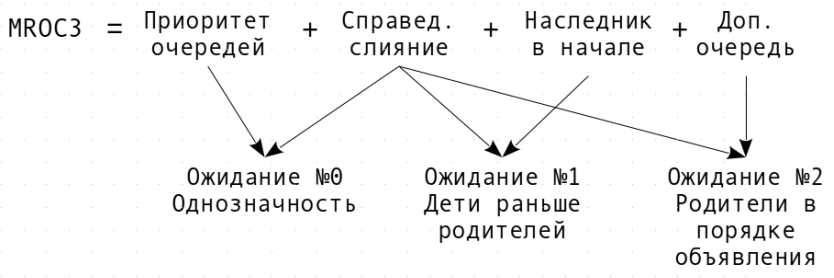


Рис. 4: Алгоритм MROC3

При визуализации на графе наследования очереди выглядят как разноцветные сосиски (начало очереди — снизу). Из этих «очередей–сосисок» классы постепенно перемещаются в итоговую очередь. При справедливом слиянии в неё могут быть перемещены только классы, стоящие в нижних концах *всех своих* очередей. Очередь считается «заблокированной», если её «нижний» класс содержится в середине другой очереди. На каждой итерации в итоговую очередь перемещается класс из первой по приоритету незаблокированной очереди.

Например, при построении линейного списка для класса G из иерархии $classA : classB(A) : classC(A) : classD(A) : classE(C, B) : classF(D, C) : classG(F, E)$: первый приоритет — у очереди класса F (см. Рис. 5A), второй — у очереди класса E (см. Рис. 5B)[4], третий — у дополнительной очереди родительских классов $[F, E]$ (см. Рис. 5C). Родительскому классу F соответствует линейный список $[F, D, C, A]$[5],

больше всех разозлились и будут пристальнее следить, чтобы лидер из низкоприоритетной очереди не встал раньше них).

[4] Приоритет очереди не связан с её местонахождением. При наследовании F стояло раньше E, то приоритет у очереди класса F выше.

[5] Строится слиянием очередей [D, A], [C, A], [D, C] и добавлением в начало класса F

родительскому классу E — $[E, C, B, A]$. В результате справедливого слияния будет построена очередь $[G, F, D, E, C, B, A]$.

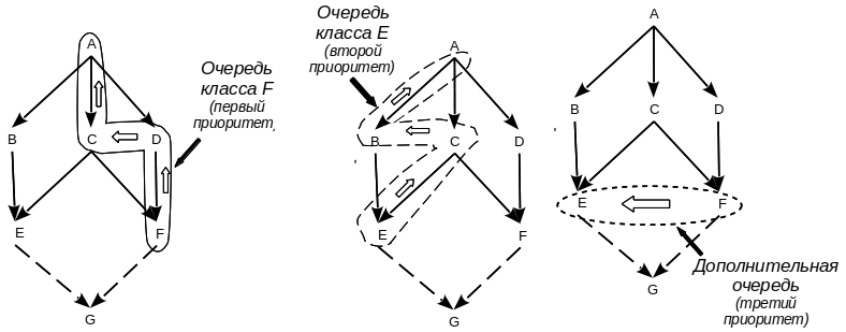


Рис. 5: A). Очередь класса F. B). Очередь класса E. C). Дополнительная очередь.

Алгоритм MROC3 строит линейный список классов, задающий порядок их просмотра для поиска поля из класса–наследника. Построение ведётся индуктивно, с помощью справедливого слияния очередей. Полученный список удовлетворяет интуитивным желаниям программиста. Для иллюстрации работы алгоритма была создана программная реализация [5].

Литература

[1] Probabilistic Inference II : `https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-034-artificial-intelligence-fall-2010/lecture-videos/lecture-22-probabilistic-inference-ii/` (дата обращения: 04.06.2021)

[2] Топологическая сортировка: `https://habr.com/ru/post/100953/`(дата обращения: 04.06.2021)

[3] Michele Simionato, The Python 2.3 Method Resolution Order, `https://www.python.org/download/releases/2.3/mro/`

[5] Строится слиянием очередей [C, A], [B, A], [C, B] и добавлением в начало класса E

[4] type.__mro__: https://pythonz.net/references/named/type.__mro__/(дата обращения: 04.06.2021)

[5] Реализация MROC3: https://gist.github.com/Derinhelm/abd10f66ae505caded35861423959945 (дата обращения: 04.04.2021)

Проскурнев Артём
Москва, ГБОУ Школа № 830

Проект: qtSimpleGraph https://github.com/temaps/qtSimpleGraph

Машинная графика в школе на свободных библиотеках

Аннотация

Школьники любят тему «Графика» на уроках информатики. Какую выбрать современную, свободную и кроссплатформенную графическую библиотеку, чтобы можно было начать с графических примитивов и научиться понимать, как работает машинная графика? Проблема всех современных библиотек заключается в написании большого количества громоздкого кода, чтобы всего лишь получить изображение простого отрезка на экране. Но теперь есть решение, которое позволяет рисовать простыми и понятными командами, вроде `Line`, `PutPixel` и тому подобными. А главное для школы то, что исполняется такой код шаг за шагом, как и учит определение алгоритма из учебников информатики.

Самой интересной темой для школьников на уроках информатики является работа с графикой. Это обусловлено тем, что данный раздел максимально приближён к желанию учащихся создавать свои компьютерные игры.

В данном докладе сделан акцент на изучение предмета «Информатика» именно в школе. Изучение программирования на начальном этапе предполагает линейные алгоритмы. Конечно, есть и ветвления, и циклы, но, в целом, программы изучаются как последовательность шагов, приводящих к результату (почти определение алгоритма в учебниках информатики). Современная концепция отрисовки графики с обновлением графического окна по таймеру не укладывается в

канву школьного программирования из-за того, что школьник ожидает возможность последовательного выполнения команд. Условно работу современных графических библиотек можно описать как создание оттиска в форме будущего рисунка, который с помощью команды обновления появится на холсте. Школьники ожидают размещение на холсте рисунков, соответствующих инструкциям, сразу после вызова. Это позволяет не хранить информацию о предыдущих нарисованных объектах.

Например, необходимо выполнить линейный алгоритм:

```
поставить точку
нажать клавишу
поставить точку
нажать клавишу
поставить точку
```

Школьник ожидает увидеть три точки после выполнения данного алгоритма.

Именно такая работа возможна с графикой в встроенных библиотеках на языках для обучения PascalABC и Кумир. Так же это реализовано и на блочном языке Scratch. Для тех школьников, которые изучают C++ или Python, такого решения не нашлось. Ставится задача найти такую современную, кроссплатформенную и обязательно свободную графическую библиотеку, которая будет работать, как предполагает концепция школьного преподавания программирования для языков C++ и Python.

К сожалению, очень хорошее решение от автора учебника Полякова Константина Юрьевича по Python[1] не решает основного вопроса с отрисовкой без специальной функции обновления холста. На сайте не указывается лицензия, поэтому нельзя предполагать, что эта библиотека является свободной. Каждый графический элемент является отдельным объектом, и если рисовать график функции точками, то будет создано огромное количество объектов-точек. По этим причинам библиотека не решает поставленную задачу. Так же, первично желателен язык C++.

Идея вернуться к библиотекам времён DOS вроде Borland BGI Graphics emulation[2], возможно, имела бы право на жизнь, если бы все такие разработки не устарели до полной безнадёжности. Ну и, конечно, это For Windows Applications...

Неплохое и легковесное решение в виде библиотеки Xcbwin[3] сначала показалось замечательной находкой. Даже была предпринята попытка её дописать и добавлен возврат кода клавиши для последующей обработки[4]. Но, после некоторого исследования, оказалось, что используемые в библиотеке решения сами сильно устарели и их поддержка будет большой проблемой в будущем.

На текущий момент, на уроках информатики используется новая, созданная в начале 2021 года, библиотека qtSimpleGraph основанная на Qt5[5]. Данная библиотека выпускается под свободной лицензией, под языки C++ и Python и эмулирует концепцию линейного рисования объектов в окне. Решение позволяет изучать машинную графику от самых основ. Первый графический примитив — точка позволяет показать как именно компьютер рисует все элементы на экране, «зажигая» конкретные пиксели конкретным цветом. Позволяет самостоятельно создать графический примитив «Линия» с использованием общего уравнения прямой и координат точек концов отрезка. Наглядно изучить определение окружности и многое другое.

В качестве лабораторных работ могут выступать простые игры типа «Змейка», «Танки», которые уже можно реализовать с использованием проекта qtSimpleGraph. Школьники с энтузиазмом изучают библиотеку и используют её для проектов по другим школьным предметам, чтобы визуализировать объекты исследования и процессы. Основы «движения» нарисованных объектов — стирание и перерисовка в новом месте очень важна для понимания детьми принципов работы компьютера. Ручные рассчёты, рисование и движение трёхмерных объектов, таких как тетраэдр или параллелепипед позволяют лучше понимать смежные предметы, такие как геометрию и тригонометрию.

В планах — создание методического пособия по проведению уроков на тему машинной графики от самых основ с использованием языков C++ и Python.

Литература

[1] Поляков К.Ю., Модуль graph для создания простых графических программ на языке Python, , https://kpolyakov.spb.ru/school/pycppbook/refs.htm

[2] Michael Main, Borland BGI Graphics and Mouse, , https://home.cs.colorado.edu/ main/cs1300/doc/bgi/bgi.html

[3] jofalk, Xcbwin — a simple C++ class for graphical outputs using XCB, https://github.com/jofalk/Xcbwin

[4] temaps, Xcbwin — a simple C++ class for graphical outputs using XCB, https://github.com/temaps/Xcbwin

[5] Проскурнев А.С., Библиотека для работы с графическими примитивами в qtCreator, 2021, https://github.com/temaps/qtSimpleGraph

Проскурнев Артем
Москва, ГБОУ Школа № 830

Проект: qtSimpleGraph https://github.com/temaps/qtSimpleGraph

Проект «qtSimpleGraph». Работа с графическими примитивами

Аннотация

Учебный проект для школ «qtSimpleGraph» представляет собой класс для отрисовки с помощью библиотеки Qt графических примитивов таких как точки, линии, прямоугольники и окружности. При подключении к программе данного класса появляется возможность отображения графических примитивов в окне приложения простыми и интуитивными командами вроде Line, SetColor, PutPixel и т. д. Это позволяет сильно снизить порог вхождения школьников в машинную графику и охватить не только профильные классы. При реализации проекта использовалась идея линейной работы с холстом, которая реализована с помощью объекта QPixmap, куда пошагово отрисовывается всё, что задаёт учащийся в программе, и «отпечатывается» на объект QPainter, что создаёт ощущение линейности выполнения. Проект активно используется на уроках и продолжает разрабатываться.

В связи с изучением в школах темы «Машинная графика» появилась необходимость в графической библиотеке для языка C++. Для языка обучения вроде PascalABC [1] или подобных встроенные возможности позволяют работать с графическими примитивами, но для языка C++ подходящее решение найти не удалось.

Задачи перед проектом поставлены следующие:

1. Языки C++ и Python;
2. GPL;

3. Последовательное мгновенное выполнение инструкций;
4. Названия функций просто описывают действие (Line, SetColor, PutPixel и т.д.)

Небольшое исследование различных решений для C++ приведено в докладе «Машинная графика в школе на свободных библиотеках» и последовательно были забракованы модуль graph Полякова К.Ю. [2], WinBGIm [3] и Xcbwin [4].

Для реализации проекта выбрана библиотека Qt, как популярная, развивающаяся и хорошо документированная. СПО qtCreator [5] позволяет создавать проекты сразу и для C++ и для Python. Основа проекта — QPainter.

Реализован класс QTSGraph в который включены графические примитивы и обработка событий мыши и клавиатуры. Для реализации четвёртой задачи по простым названиям функций, стандартное событие paintEvent вызывает событие PaintBox, которое вынесено в файл main.cpp с основным кодом и основной функцией main. Весь основной код работы с графикой пишется в реализации метода QTSGraph::PaintBox, что позволяет вызвать функции отрисовки примитивов просто по имени и использовать всю структуру класса буквально изнутри. Получается, что проект не является независимой библиотекой, но для удобства работы, исследования внутренней структуры и постепенного перехода к прямой работе с современной концепцией работы с графикой, проект намеренно развивается именно в этом направлении.

Минимальная программа с рисунком main.cpp выглядит так:

```
#include "qtsgraph.h"
int main(int argc, char *argv[])
{
    QApplication a(argc, argv);
    QTSGraph w(601, 601);
    w.show();
    return a.exec();
}
void QTSGraph::PaintBox()
{
    SetColor(clRed);
    Circle(300, 300, 300);
}
```

Результатом будет квадратное окно размером 601 на 601 пикселей в центре экрана с вписанной в него красной окружностью.

Для решения третьей задачи последовательного мгновенного выполнения инструкций без «заморозки» окна во время, например, паузы используется способ «прокрутить» ждущие события, пока окно рисования «стоит на паузе»:

```
QCoreApplication::processEvents(QEventLoop::AllEvents, 50);
```

Это позволяет создать такие функции как «пауза», «ждать клика мыши», «ждать нажатия клавиши» и т. п. Так же это влечёт некоторые проблемы с закрытием программы. Например, если просто закрыть окно приложения во время выполнения цикла с условием выхода, например, while(KeyPressed()), то процесс не завершается, поэтому была добавлена команда exit(0) в событие QTSGraph::closeEvent.

Сама же отрисовка производится на объект QPixmap Canvas, который просто «отпечатывается» потом на холст окна:

```
QPainter p(this);
p.drawPixmap(0, 0, Canvas);
```

Это позволяет не терять нарисованное ранее, а либо продолжать рисовать, либо рисовать поверх уже имеющегося.

Документация проекта в Wiki содержится на GitHub[6]. Созданы разделы по всем графическим примитивам, которые программа умеет отображать. Проект продолжает разрабатываться и активно использоваться на уроках информатики в школе. Приветствуется помощь и участие в разработке.

Литература

[1] Михалкович С. С., Бондарев И., Погорелов А., Система программирования PascalABC.NET, http://pascalabc.net/stati-po-pascalabc-net/obuchenie-programmirovaniyu/16-grafika-

[2] Поляков К. Ю., Модуль graph для создания простых графических программ на языке Python, https://kpolyakov.spb.ru/school/pycppbook/refs.htm

[3] Michael Main, Borland BGI Graphics and Mouse, https://home.cs.colorado.edu/~main/cs1300/doc/bgi/bgi.html

[4] jofalk, Xcbwin - a simple C++ class for graphical outputs using XCB, https://github.com/jofalk/Xcbwin

[5] Qt Group, Qt Creator, https://www.qt.io/download-qt-installer

[6] Проскурнев А. С., Документация проекта qtSimpleGraph, https://github.com/temaps/qtSimpleGraph/wiki

;

Турчанский Никита, Проскурнев Артём
Москва, ГБОУ Школа № 830

Проект: Снап! http://snap.school830.ru,
https://github.com/turkannnn/

Адаптация перевода Системы Наглядного Анимационного Программирования для внедрения её в школьную программу обучения начальных классов

Аннотация

Выполнен перевод Системы Наглядного Анимационного Программирования на русский язык[1]. Снап! может применяться в образовательных и практических целях в качестве системы для разработки и замены Flash-уроков из приложений к учебникам младших классов. Также сделан справочник «МойСнап![2]», в котором находится основная информация о работе с системой, несколько готовых уроков для обучения основам программирования и возможность запуска уроков. В будущем планируется добавить возможность запуска уроков, созданных в рамках данной системы, в сценариях МЭШ.

Цель данного проекта заключается в адаптации для учителей и учеников начальной школы перевода Снап! — бесплатного, образовательного, блочного языка виртуального программирования, разработанного на основе Scratch в Калифорнийском университете в Беркли[1].

[1] http://snap.school830.ru
[2] http://snap.school830.ru/mysnap

Основной идеей проекта является его применение в образовательных и практических целях в качестве системы для разработки и замены Flash-уроков из приложений к учебникам младших классов. Справочник с начальными сведениями о работе с системой, который был сделан в виде сайта, может помочь начинающим пользователям и учителям.

Так как Снап! имеет лицензию GPLv3, все изменения принятые в целях выполнения данного проекта полностью свободны и рекурсивно лицензируются по GPLv3.

Полное прекращение поддержки Flash от Adobe[2] является очень весомой причиной поиска альтернатив. Самой удачной альтернативой можно считать данный проект. Подавляющее большинство учебных программ на дисках, прилагающихся к учебникам, сделаны с помощью технологии Flash, и большинство учителей уже не сможет пользоваться этими программами. С использованием проекта Снап! можно делать такие же обучающие программы. Основной идеей является формат, когда дети 4–5 класса могут помогать учителям делать программы для более младших классов.

На данный момент аналогами Системы являются Scratch[3] и Логоподобные языки (Логомир, Перволого и т.д.). Они составляют промежуточное звено между языками программирования дошкольного уровня, такими как Blockly и ПиктоМир[4], и языками программирования высокого уровня, такими как C и Pascal.

Проблема Логомиров и Перволого заключается в том, что они работают на пиктограммах. Во втором классе уже следует переходить на программирование словами. Когда алгоритм составлен словами, то уже проще перейти к любому языку. Все алгоритмы в Снап! строятся из слов записанных в блоках, что позволяет учащимся избежать ошибок в синтаксисе.

Проблема языка Scratch заключается в лицензии MIT. Данная лицензия позволяет в любой момент сделать коммерческую программу закрытой и не разрешает скачивание и модифицирование исходного кода при не соблюдении поставленных условий. Также в Scratch, в отличие от Снап!, никак не реализована возможность объяснения того или иного блока с командой, что тоже является минусом.

Важной частью работы является перевод системы. В первую очередь требовалось перевести текст. В основном весь текст находится в блоках с командами и в разных меню, начиная от меню регистрации аккаунта и заканчивая меню создания блока. На данный момент ра-

бота с переведением текста вручную завершена, но часто с выходом обновлений добавляются новые возможности, блоки и, соответственно, новый текст. Я это отслеживаю и в ближайшее время вношу и изменения.

Также переводу подлежали вспомогательные картинки, о которых я упомянул ранее. Каждый блок имеет свою справку. Справка выполнена в виде картинки, где написано: что блок делает, для чего используется. Также приведены примеры его использования. На данный момент переведены не все картинки, но те, которые уже переведены, находятся на сервере, и я постепенно добавляю туда новые. Стоит сказать, что все картинки были переведены и перерисованы вручную.

Хотелось бы рассказать о справочнике, который был сделан в виде сайта и назван «МойСнап!». В основном этот справочник предназначен для пользователей, которые впервые пользуются Снап!. В справочнике находится основная информация о работе с системой, несколько готовых планов для проведения уроков по основам программирования и возможность запуска созданных программ.

В будущем, помимо окончания перевода картинок, я планирую добавить возможность запуска работ Снап! в сценариях МЭШ. В самой системе такой функционал уже реализован с помощью функции в ранее упомянутом справочнике.

Литература

[1] Jens Mönig, Snap! Build Your Own Blocks, `https://snap.berkeley.edu/`

[2] Adobe Systems Software Ireland Ltd, Adobe Flash Player, `https://get.adobe.com/ru/flashplayer/about/`

[3] MIT, Scratch — Imagine, Program, Share, `https://scratch.mit.edu/`

[4] ИнфоМир, ПиктоМир, `https://piktomir.ru/`

Екатерина Орлова, Иван Хахаев
Санкт-Петербург, СПбГЭТУ «ЛЭТИ»
https://github.com/KatiaOrl/LibreOfficeExtensions

Актуализация расширения BasicAddonBuilder для современных версий LibreOffice

Аннотация

Рассматривается организация пользовательских расширений в LibreOffice и проблемы в автоматизации создания таких расширений в современных версиях пакета. Описываются ошибки, приводящие к некорректной работе расширения BasicAddonBuilder и пути их исправления.

Одна из ключевых полезных особенностей LibreOffice — возможность пользователям создавать библиотеки макросов и публиковать их в виде расширений, которые могут быть легко установлены другими пользователями с использованием штатных средств LibreOffice («Сервис/Управление расширениями»). Расширение может быть создано из пользовательской библиотеки макросов или темы галереи путём редактирования нескольких файлов в подкаталогах расширения:

- `manifest.xml` в подкаталоге `META-INF`, содержит метаданные расширения (относительный путь к файлам расширения, расположение служебных файлов);
- `pkg-description.txt` в подкаталоге `description` или `pkg-desc` (у разных расширений по-разному), содержит текст с описанием расширения;
- пиктограмма расширения (файл `png`) для встраивания в панели инструментов в подкаталоге `images`;
- файл с текстом лицензии (`license.txt`) в подкаталоге `registration` или `licenses` (у разных расширений по-разному), содержит текст с авторской лицензией;
- `Addons.xcu` и `descriprion.xml` в основном каталоге расширения. Первый файл содержит меню расширения, второй — информацию о расширении (уникальный идентификатор, версию, зависимости).

Файлы расширения (модули на LibreOffice Basic (`*.xba`) и диалоги (`*.xdl`) должны размещаться в каталоге, имя которого соответсвует имени расширения.

Для автоматизации процесса создания расширений ещё для ранних версий OpenOffice.org было создано расширение BasicAddonBuilder (2006–2008, Paolo Mantovani), которое развивалось и для LibreOffice вплоть до версии LibreOffice 5.2.7, после чего авторская поддержка расширения была прекращена (или приостановлена).

В текущих версиях LibreOffice (7.0.x и 7.1.x) при попытках использования расширения BasicAddonBuilder выявлены следующие ошибки:

1. При запуске расширения в LibreOffice 7 работа приложения может аварийно завершиться;
2. Если исходная библиотека макросов создана как встроенная в текущий документ, то при сохранении расширения (нажатие на кнопку `«Create»` на последнем шаге диалога) макрос завершается с ошибкой:

   ```
   Type:   com.sun.star.ucb.InteractiveAugmentedIOException
   Message: a file or directory could not be deleted.
   ```

   Файл расширения в результате не создаётся;
3. По окончании работы BasicAddonBuilder приложение LibreOffice аварийно завершает работу, расширение создаётся;
4. При создании новой панели инструментов и выборе пункта `«Compatible with AOO 4.0.0 and higher versions»`, панель инструментов при установке готового расширения не добавляется.

В результате анализа кода выявлены и устранены причины этих ошибок (см. таблицу 1).

Версия BasicAddonBuilder, корректно работающая в LibreOffice 7, опубликована на GitHub[1].

[1] `https://github.com/KatiaOrl/LibreOfficeExtensions`

Было	Стало
Модуль DlgWizard_step01 функция EnableStepControls()	

Обращение к необъявленной структуре
`g_oDlgWizardModel.Roadmap`
	Объявлено явно
	`Dim oRoadmapModel As Object`
	`oRoadmapModel = g_oDlgWizardModel.Roadmap`

`For I=1 To g_oDlgWizardModel.Roadmap.Count-1`
`g_oDlgWizardModel.Roadmap(I).Enabled = bEnable`
`Next I`
	`For I = 1 To oRoadmapModel.Count - 1`
	`oRoadmapModel(I).Enabled = bEnable`
	`Next I`

Модуль PkgExporter1 процедура ExportDialogLibrary	
Вызов функции удаления незакрытого временного файла `oFileAccess.kill(sTempDocUrl)`	Вызов закомментирован, файл не удаляется, а переписывается
Модуль DlgWizard_main функция ShowDialog()	
Вызов `{g\_oDlgWizard.dispose` с последующим вызовом `endExecute`	Вызов `g_oDlgWizard.dispose` закомментирован для устранения двойного освобождения ресурсов
Меню создания панели инструментов нового расширения	
Два пункта: • Compatible with AOO 4.0.0 and higher versions • Compatible up to AOO 3.4.1	Для LibreOffice 5.2+ всё совместимо с AOO 3.4.1, поэтому меню изменено • Compatible with AOO 4.0.0 and higher versions • Compatible up to AOO 3.4.1 or LO 5.2.7 and higher

Таблица 1: Изменения в коде BasicAddonBuilder

Яков Шпунт
Москва, Comnews Group, онлайн-проект Vision, журнал «Стандарт»

САПР и Linux. Проблемы и решения. Советы постороннего

Аннотация

В технических вузах и ссуз САПР является одним из ключевых предметов. В силу ряда причин наиболее распространено изучение продуктов Autodesk, прежде всего, AutoCAD. Но это ПО в среде Linux не работает, что часто оказывается непреодолимым тормозом для внедрения СПО в учебном заведении. Тем не менее, есть и альтернативы и обходные пути.

В ноябре прошлого года стало известно о том, что один из дистрибьюторов Microsoft отказал МГТУ им. Баумана в продлении лицензий на Windows. Это оказалось для одного из ведущих технических вузов страны очень неприятным сюрпризом. Особенно сложным оказалось найти альтернативу ПО САПР, прежде всего, от Autodesk. Тем более, что МГТУ стал по факту одним из главных центров компетенции и обучения AutoCAD в России.

В схожей ситуации может оказаться практически любой технических вуз в России. Так что нужно быть готовыми к тому, что придётся искать альтернативу привычному ПО и быстро адаптировать учебные программы. Может также вмешаться и третья сила, а именно потенциальные работодатели выпускников, которые заинтересованы в том, чтобы будущие работники умели использовать то ПО, которое применяется у них или будет применяться после реализации программ импортозамещения. Вместе с тем, есть и проблемные направления, где найти альтернативу продуктам Autodesk пока трудно.

Блеск и нищета AutoCAD

AutoCAD занял своё место в образовательном сегменте благодаря двум обстоятельствам. Прежде всего, это был первый CAD для платформы PC, который появился на российском рынке и был для него адаптирован (локализован, соответствовал российским ГОСТ для подготовки техдокументации). При этом Autodesk был весьма активен в продвижении своих продуктов для сферы образования. Ведущие вузы получали его ПО практически бесплатно. В итоге в довольно короткий срок появились качественные учебные материалы, позволявшие освоить продукт «с нуля» всем желающим.

Однако AutoCAD был не лучшим продуктом на рынке. Это хорошая система для подготовки чертежей, но этого недостаточно для проведения всех необходимых расчётов (прочностных, тепловых и т. д.), а также подготовки изделия к производству. С её помощью можно добиться лишь базового уровня зрелости среды проектирования согласно классификации аудиторской компании Pricewaterhouse Coopers (PwC)[1]. Ни о какой единой цифровой модели готового изделия, где реализованы все этапы его жизненного цикла, от концепции до вывода из эксплуатации, тут речи быть не может. Необходимо дополнять AutoCAD другими продуктами от сторонних производителей, часто дорогостоящими, например, CATIA.

В итоге AutoCAD в российском машиностроении малопопулярен. Например, из всех предприятий Воронежа он используется лишь на одном. В целых отраслях, например, судостроении, авиационной или космической фактическим стандартом являются другие продукты, многие из которых работаю и в Linux.

[1] `https://www.pwc.ru/ru/assets/prop-tech-2020.pdf`

Рис. 1: Уровни зрелости среды проектирования по PwC.

Коммерческие альтернативы

У AutoCAD существует не менее 4 коммерческих альтернатив, которые работоспособны в Linux. При этом как минимум две из них (NX и Компас) стали стандартом в ряде отраслей, а BricsCAD бесплатен для некоммерческого использования.

Раннее предлагалась бесплатная версия `Dassault DraftSight`, но в 2019 году вендор от неё отказался. Также прекращён выпуск нативной Linux-версии этого приложения. Хотя до этого именно данный продукт был одним из наиболее активно рекомендуемым решением для тех, кто планировал миграцию в среду Linux и при этом требовалась работа с приложениями САПР.

`BricsCAD` от бельгийской компании Bricsys официально бесплатен для сферы образования. Надо, однако, иметь в виду, что срок действия академической лицензии ограничен одним годом, и его нужно регулярно продлевать. Есть также облачная версия.

Для BricsCAD существует более 400 приложений, позволяющих использовать его в архитектуре, строительстве, машиностроении, проектировании инженерных сетей, электрике, автоматике, ГИС. Данная САПР полностью совместима с AutoCAD как по формату файлов, так и по набору инструментов и интерфейсам для адаптации

и разработки. BricsCAD работает в Windows, macOS и Linux. Полностью соответствует российским стандартам. Старшие версии, помимо CAD, включают также CAM и CAE модули. С их помощью можно добиться уже как минимум 2 уровня зрелости среды проектирования, а то и третьего.

Ещё одной альтернативой является «Компас 3D» от российской компании «Аскон». Есть бесплатная учебная версия, предназначенная для сферы образования, при этом обладающая полной функциональностью. Несмотря на то, что формально нативной версии для Linux нет, продукт работоспособен через Wine, при этом такой вариант официально поддерживается, что бывает довольно редко. Имеется программный комплекс САПР «Макс», работающий без использования средств виртуализации. Данное решение, ориентированное на проектирование трубопроводных сетей, является первым в своём роде, надеюсь, за ним последуют и другие.

Это самодостаточная полнофункциональная САПР, полностью соответствующая концепции сквозного проектирования. Полностью соответствует ГОСТ 2.052-2015 «Электронная модель изделия». Есть внедрения в таки отраслях как машиностроение, приборостроение, авиастроение, судостроение, станкостроение, вагоностроение, металлургия, промышленное и гражданское строительство, производство товаров народного потребления. Поддерживается экспорт и импорт документов в форматах как AutoCAD, так и Siemens NX.

Siemens NX. Один из лидеров рынка САПР. Фактический стандарт для очень многих отраслей. Применяется всеми ведущими российскими авиационными КБ, ГК «Роскосмос», КАМАЗ.

Уже давно есть нативная версия для Linux. Функциональность является одной из самых полных на рынке. Содержит не только инструменты CAD/CAM/CAE, но и одну из лучших на рынке систем поддержки жизненного цикла изделия (PLM). Курс обучения NX уже давно читается в ведущих вузах авиационного профиля, в частности, МАИ. Однако продукт дорогостоящий и сложный. Переработка учебного курса с AutCAD на NX будет трудоёмкой и длительной.

IBM CATIA. Продукт с длительной историей (первая версия вышла ещё в 1981 году), несколько раз продавался и покупался. В нём впервые была реализована концепция PLM. Функциональность в целом та же, что и главного конкурента Siemens NX. Главная сильная сторона — наиболее развитые на рынке средства взаимодействия внутри команд проектировщиков. Поддерживаются все основные операционные

системы, включая Linux. Стоит иметь в виду, что на рынке сосуществует две версии, имеющие серьёзные различия и ориентированные на разные сегменты рынка.

Свободные САПР

Имеется два проекта, более-менее пригодные для практического использования. `QCad Community Edition` и его форк `LibreCAD` наиболее известны, но при этом и наименее функциональны. Однако в ряде стран данные приложения чрезвычайно популярны. Например, на Тайване по числу инсталляций они лидируют.

Эти приложения являются чистыми CAD, средства CAM/CAE отсутствуют. Да и функциональность CAD лишь базовая и серьёзно уступает лидерам рынка. Вместе с тем, продукт вполне подходит для начального обучения. В связке с Dia или LibreOffice Draw является полноценной заменой продуктов вроде Visio Technical.

Наиболее функциональным является `FreeCAD` (не путать с `freeCAD`). Продукт существует с 2002 года, но релизная версия до сих пор не выпущена. Тем не менее, продукт вполне работоспособен. Его функциональность уже сейчас находится на уровне CATIA 5. Как и лидирующие системы, использует модульную архитектуру. Легко интегрируется с другими приложениями. Совместим со многими популярными САПР приложениями, включая 3DS, AutoCAD, NX. Вполне работоспособный продукт, хотя и с не вполне стандартным интерфейсом и приёмами.

BIM. Главная проблемная точка

В 2021 году использование технологии BIM (Building Information Modeling или Building Information Model — информационное моделирование здания или информационная модель здания) стало обязательным для объектов, возводимых с использованием бюджетного финансирования. И это не случайно: под данным исследования PwC, срок подготовки проектной документации уменьшаются на 10%, при этом количество ошибок проектирования снижается на 10%. Если же не ограничиваться применением BIM только фазой проектирования, то выигрыш будет ещё более впечатляющим. Так, по данным Минстроя России, сокращение времени на проектирование может снижаться и на 50%, количество ошибок и погрешностей в документации

на 40%, а затраты на строительство и эксплуатацию на 30%. Соответственно, растёт востребованность специалистов, которые владеют навыками построения BIM-моделей. Причём эти разработчики должны быть компетентными и в других отраслях. «К нам приходит множество проектировщиков, которые хотят устроится на работу. Мы заметили, что нередко за знанием проектировщиком других современных программ скрывается его техническая некомпетентность. Поэтому чертежи в 3D выглядят у этих проектировщиков очень эффектно, но совершенно безграмотно с технической точки зрения», — жалуется руководитель мастерской ООО «Траст Инжиниринг» Александр Иванов.

Наиболее популярным в России инструментом BIM-моделирования является Autodesk Revit. Как показало исследование аналитического центра ComNews, практически у всех участников рынка информационного моделирования зданий данное ПО являлось основным. Естественно, строительные и архитектурные вузы вынуждены следовать за потенциальными работодателями своих выпускников.

При этом Revit выпущен только под Windows. И что делать, если данное обстоятельство стало препятствием для миграции вуза на Linux? Есть два варианта. Это или использование VDI, или выбор другого продукта. Тут возможно два варианта: FreeCAD, у которого имеется соответствующий модуль, или российского продукта Renga. Ведь импортозамещения тоже никто не отменял, и Минстрой России обещает даже данный проект ускорить.

Литература

[1] Малышев О., Данченок А., PropTech в России: Обзор практики применения BIM-технологийи инновационных решений в области проектирования. https://www.pwc.ru/ru/assets/prop-tech-2020.pdf

[2] Шпунт Я., BIM. Испытание на прочность https://www.comnews.ru/content/212032/2020-12-14/2020-w51/bim-ispytanie-prochnost

[3] Смирнова Н., Выборочный опрос участников рынка информационного моделирования зданий https://www.comnews.ru/sites/default/files2019/styles/large/public/reviews/2020-12/678567567.png?itok=2jICspvx

Диденко Денис
Щёлково, ГБПОУ Московской области «Щёлковский колледж»

Свободное программное обеспечение для мониторинга и администрирования компьютерного класса

Аннотация

В докладе рассмотрен собственный 3-летний успешный опыт работы с прложением Veyon для управления компьютерным классом.

Общие сведения

Для проведения занятий по информатике и информационным технологиям для мониторинга и управления компьютерным классом в ГБПОУ МО «Щёлковский Колледж» СП-5 используется программное обеспечение Veyon (продолжение хорошо зарекомендовавшей себя программы Italc).

Организация работы со свободным приложением Veyon для мониторинга и администрирования компьютерного класса в Щёлковском филиале ГБПОУ МО «Щёлковский Колледж»

В ГБПОУ МО «Щёлковский Колледж» уже около 3 лет довольно успешно на компьютерах студентов используется приложение Veyon для мониторинга и администрирования компьютерного класса. За это время были апробированы несколько видов подобных приложений, но окончательный выбор был сделан в даного. Он обусловлен рядом причин:

1. Оконная и полноэкранная демонстрация своего рабочего стола на один компьютер или весь класс, что практически исключает необходимость покупки в кабинет проектора;
2. Встроенная поддержка протокола LDAP/AD;
3. Ведение мониторинга и дистанционное управления всеми компьютерами учеников;

4. Снятие скриншотов;
5. Возможность блокировки отдельного компьютера или всего класса;
6. Отправка текстовых сообщений;
7. Управление питанием компьютера (включение, выключение и перезагрузка компьютеров учащихся);
8. Удалённое выполнение скриптов и команд;
9. Поддержка VPN, что позволяет присоединиться к урокам с домашнего компьютера;
10. Возможность удалённого просмотра рабочего стола без управления;
11. Создание классов и компьютеров в классах;
12. Открытие адреса веб-сайтов на удалённых компьютерах.

В ГБПОУ МО «Щёлковский колледж» СП 5 в двух кабинетах информатики используется данная программа и вызывает у преподавателей прекрасные референции. На компьютерах установлены две операционные системы Windows (для работы с некоторыми проприетарными программами) и Linux с дубликатами электронных ключей данной программы, что позволяет мониторить класс независимо от работающей операционной системы.

Как видно, для бесплатного приложения — Veyon обладает мощным функционалом, способным конкурировать с платными аналогами.

Выводы

По результатам апробации приложения Veyon был сделан однозначный вывод, что процесс обучения в колледже довольно успешно реализуется на его основе независимо от установленной операционной системы.

Используя данное программное обеспечение можно успешно вести управление компьютерным классом, отправлять сообщения, блокировать компьютеры (полезно, чтобы обратить внимание учащихся на нужную задачу в настоящее время), управлять удалёнными компьютерами — одним словом практически всё то, за что другие программы просят денег.

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.

Максим Каторгин, Алексей Ерпелев, Денис Селютин, Владимир Симонов
Москва, Российский государственный социальный университет

Программирование с использованием СПО, электроники и робототехники как средство развития координации, моторики и реакции для лиц с заболеванием «Детский церебральный паралич»

Аннотация

В последние годы значительно растёт количество методик и способов реабилитации лиц с детским церебральным параличом. Представленная в данной статье методика подходит лицам с приобретённым ДЦП, поскольку в этом случае сохранён интеллект, что важно в процессе реабилитационной работы. Используемая программно-аппаратная база (на основе СПО) позволяет развивать поэтапно как навыки программирования, так и мелкую моторику, а собственно занятия и получение результата вызывает у детей искренний интерес достаточно продолжительное время, что важно для положительной динамики реабилитации.

Детский церебральный паралич — группа хронических симптомокомплексов двигательных нарушений, достаточно широко распространённых в настоящее время. Нарушение двигательной способности здесь имеет место из-за поражения мозга. Чтобы заболевание не прогрессировало, необходимо поддерживать форму путём выполнения специально назначенных физических упражнений, развития мелкой моторики [1].

Существующие в настоящее время методики эффективны, однако представляют собой скучное, монотонное и однотипное занятие, которое достаточно быстро надоедает и забрасывается ребёнком. Только большие прилагаемые волевые усилия со стороны ребёнка и родителей позволяют продолжать занятия и не допустить остановки [2].

Авторами предложена методика [3], сущность которой состоит в использовании в качестве тренажёров стартовых наборов, основная основополагающая цель которых — получение навыков программирования, электроники и робототехники. Конструирование с использованием этих тренажёров вызывают у детей искренний интерес в течение достаточно продолжительного времени. Наборы распределены по возрастам, и с помощью данных наборов ребёнок не только разовьёт навыки мелкой моторики, но и сможет познакомиться с основами программирования, электроники и робототехники.

Предлагаемая методика состоит в том, что лицам, страдающим ДЦП, предлагается в качестве тренажёров для развития мелкой моторики использовать различные наборы в вышеуказанной области электроники, программирования и робототехники. Последовательность занятий выстраивается так, чтобы постепенно увеличивалась точность движений. Указанное достигается тем, что сначала занятия проводятся с крупными деталями, далее осуществляется переход к деталям среднего размера, а затем — к более мелким.

Важнейшим преимуществом здесь является, что, во-первых, ребёнок активно интересуется результатом, во-вторых, вовлечён в образовательный процесс и получает ценные навыки в области программирования и электроники.

Разумеется, здесь не может быть общего, универсального временного плана занятий, здесь всё индивидуально, и занятия выстраиваются в соответствии с индивидуально достигнутыми результатами [4].

Ниже представлены рекомендуемые этапы, составляющие предлагаемую методику.

Для первого этапа занятий хорошо подойдёт электромеханический конструктор LEGO Education Mindstorms EV3. Мозгом набора является программируемый микрокомпьютер EV3, набор содержит различные моторы, датчики и конструктивные элементы LEGO. Набор состоит из достаточно крупных деталей, легко поддающихся захвату кисти. Управление исполнительными элементами осуществляется с помощью микрокомпьютера на языке MicroPython, который является упрощённой версией одного из самых популярных в мире языков программирования.

Для следующих этапов возможно использование стартового набора Arduino, детали в котором гораздо мельче, что способствует положительной динамике развития функциональности кисти. Вычислительной платформой здесь является микроконтроллер Arduino

Uno [5]. Работа с набором предназначена для детей более старшего возраста (с 12 лет), так как требует определённых знаний физики, знакомства с языком программирования C++ (версия Wiring), средой программирования Arduino IDE, а также изучения основ программирования микроконтроллеров. При этом ребёнок создаёт свои первые настоящие электронные устройства. Для ребят младше, которые прошли первый этап, но не имеют базовых знаний физики, промежуточным этапом может стать электронный конструктор Микроник. Руководство для данного конструктора написано в занимательном стиле и подбадривает на прохождение новых экспериментов.

Детали набора Arduino также подходят к другим микроконтроллерным наборам, поэтому для дальнейшего совершенствования навыков программирования может использоваться платформа Iskra JS, имеющая сходство с Arduino по функционалу.

Результаты и выводы. Предлагаемая методика была протестирована одним из соавторов работы на самом себе, и получены следующие результаты:

- остановилось прогрессирование атетоза обеих рук (засвидетельствовано при периодических врачебных осмотрах);
- заметно улучшились способности к мелкой моторике, что отмечено и самим автором, и его коллегами;
- увеличилась скорость сборки и качество работы с мелкими компонентами проектов.

Таким образом, методика доказала свою работоспособность, эффективность и авторами предполагается дальнейшее еѐ развитие.

Литература

[1] Немкова С. А. и др. Детский церебральный паралич: диагностика и коррекция когнитивных нарушений: учеб.-метод, пособие / М-во здравоохранения и соц. Развития Российской Федерации, Науч. центр здоровья детей РАМН, Российский нац. исслед. мёд. ун-т им. Н.И. Пирогова; — М.: Союз педиатров России, 2012. — 60 с.

[2] Козявкин В. И., Бабадаглы М. А., Лунь Г. П. и др. (всего 7 соавторов). Система интенсивной нейрофизиологической реабилитации — метод Козявкина. Пособие реабилитолога. / Львов. Изд-во «Дизайн-студия «Папуга»», 2012. — 240с. URL: `http://sensint.ru/sites/default/files/newsinr_book_ru.pdf`. (дата обращения: 07.05.2021).

[3] Селютин Д. Ю., Каторгин М. К., Симонов В. Л. Электроника, программирование и робототехника как вид реабилитации с тренировкой моторной реакции для лиц с заболеванием «Детский церебральный паралич» / В сборнике: Современные информационные технологии в образовании, науке и промышленности. Искусственный интеллект в создании картин. Сборник трудов XVIII Международной конференции и XVI Международного конкурса научных и научно-методических работ. Ответственный редактор и составитель Т.В. Пирязева. Москва, 2021. С. 87–89.

[4] Разработка прототипа робота-помощника для лиц с ограниченными возможностями здоровья на базе промышленного робота-манипулятора KUKA с электрическим захватом под управлением платформы Arduino и СПО / Каторгин М., Михайлов Н., Ерпелев А., Бакалым И., Улатов В., Симонов В. / В кн: Свободное программное обеспечение в высшей школе. Сборник тезисов XV конференции. Отв. редактор В.Л. Чёрный. 2020. С. 109–112.

[5] Сайт, посвящённый разработке схем в области электроники, программирования и робототехники / Режим доступа: URL: `https://amperka.ru/product/mikronik` (дата обращения: 07.05.2021).

Максим Каторгин, Алексей Ерпелев, Ли Юньхань, Денис Селютин, Владимир Симонов

Москва, ФГБОУ ВО «Российский государственный социальный университет», ФГБОУ ВО «Московский авиационный институт (национальный исследовательский университет)»

`https://github.com/MaksimKatorgin/Meteo/blob/main/meteo-norm-to-csv.py`

Программно-аппаратный мониторинг состояния внутренней среды специализированного помещения

Аннотация

Разработано программно-аппаратное средство для анализа изменений основных параметров в помещении: освещённости, атмосферного давления, влажности и температуры, с целью оценки комфортности условий труда на рабочем месте. Основа разработки — одноплатный компьютер Raspberry Pi 4B, Troyka Cap и сенсоры. Разработка осуществлена с использованием СПО.

Контроль состояния внутренней среды места обитания человека, являются одним из решающих факторов сохранения здоровья человека, независимо от того, относится ли это к жилым производственным или иным типам помещений [1]. Следовательно, контроль состояния среды обитания должен производиться везде, где находится человек — жилые, производственные помещения, кабины и салоны летательных аппаратов, здания и залы аэропортов, и т. д.

Для контроля климатических параметров состояния внутренней среды помещений разработано достаточно много устройств: люксметры, термометры, барометры, гигрометры и т.д. [2], [3]. Использование всех без исключения устройств является технически невыполнимой задачей, и на практике чаще всего используют набор сенсоров, преследующих цель автоматического определения каких-то частных задач, например, сертификации жилого помещения, офиса, спортивных комплексов, помещений аэропортов и т. д.

Целью настоящего проекта является разработка устройства мониторинга и анализа основных параметров помещения — освещённости, влажности, температуры и атмосферного давления.

Существует достаточно большое разнообразие программно-аппаратных средств, которые можно использовать для достижения поставленной цели — одноплатный компьютер Raspberry Pi 4B, Arduino-совместимые компоненты, сенсоры, исполнительные устройства и т. д. [4].

Достаточно удобным является использование платы Troyka Cap, установленной на контакты GPIO одноплатного компьютера Raspberry Pi. При этом с целью измерения выбранных параметров возможно использование необходимых Troyka-модулей, а именно — датчика освещённости (основа — фоторезистор тип GL5528), цифровой датчик температуры и влажности (сенсорная сборка KY-015), цифровой датчик барометр v1 (основа — чип LPS331AP от STMicroelectronics) [5].

Для Rasberry Pi B родным является язык Python, под который имеются необходимые библиотеки для подключения всех компонент устройства. Указанное определило использование языка Python. В качестве среды разработки использовалось среда IDE Thonny.

Для работы устройства необходим источник постоянного тока 5А, что заметно больше, чем для предшественника — Raspberry Pi 3.

Устройство определяет дату, время, считывает показания с датчиков, определяет сезон, проверяет соответствие нормам СанПин [1], выводит данные на экран и заносит в электронную таблицу.

Разработанное устройство выполнено в мобильном варианте и было успешно протестировано для различных типов помещений.

Литература

[1] Постановление Главного государственного санитарного врача Российской Федерации от 10 июня 2010 г. N 64 г. Москва «Об утверждении СанПиН 2.1.2.2645-10». URL: https://rg.ru/2010/07/21/sanpravila-dok.html (дата обращения 14.05.2021). — Текст : электронный.

[2] Каторгин М. К. Портативная метеостанция на базе платы Raspberry PI. — В сборнике: Сборник материалов IV Всероссийской научно-практической конференции магистрантов. 2019. С. 59-63.

[3] Использование СПО в учебном процессе на примере разработки портативной метеостанции на Raspberry Pi / Ерпелев А., Симонов В. / В книге: Свободное программное обеспечение в высшей школе. Сборник тезисов XV конференции . Отв. редактор В. Л. Чёрный. 2020. С. 151–155.

[4] Применение технологий умного дома в интернатах и других организациях социальной сферы / Бакин М. А., Венина Е. Б., Дербышева А. М., Ерпелев А. В., Овсянников Н. А., Петрова Е. А., Рубанкова А. П., Симонов В. Л. В сборнике: Богатство России. Сборник докладов. 2019. С. 156–157.

[5] Амперка: официальный сайт. — Москва. URL: https://amperka.ru/ (дата обращения: 14.05.2021). — Текст : электронный.

Наталья Хамидуллина, Алексей Ерпелев, Максим Каторгин, Денис Селютин, Владимир Симонов
Москва, Российский государственный социальный университет

Сравнительный эксперимент функционирования автоматизированной системы саморегулирования режима аквариума, созданной с помощью свободного программного обеспечения, и нагревателя, управляемого встроенным термостатом

Аннотация

В работе рассмотрен эксперимент, использующий платформу Arduino для управления системой регулирования температурным режимом аквариума. Приведено сравнение основных показателей, на основе данных сенсоров, необходимых для обеспечения автоматического регулирования. В разработке использовалось свободное программное обеспечение.

Факторы, влияющие на микроклимат, можно разделить на две группы: нерегулируемые (комплекс климатообразующих факторов) и регулируемые (интенсивность теплового излучения от нагревательных приборов, кратность воздухообмена, количество растений и животных в аквариуме).

Для проведения экспериментов была выбрана температура в качестве первого параметра, реализуемого автоматизированной системой. Температура, бесспорно, является самым важным параметром для каждого живого организма, это касается и аквариума. Кроме того, температура это однозначный параметр, относительно быстро меняющийся в течение дня (поэтому экспериментальное время не растягивается слишком долго), а приборы контроля и управления достаточно дёшевы, кроме того, имеется большое разнообразие датчиков и исполнительных устройств на рынке.

Микроконтроллерная платформа Arduino имеет много возможностей, с помощью которых можно реализовать функцию поддержания температуры. Плата Arduino Uno представляет собой открытую платформу, позволяющую собирать разнообразные устройства. Кроме того, Arduino Uno может работать как в связке с компьютером, так и автономно.

Основные достоинства указанной платформы — малый размер, низкое энергопотребление, свободное программное обеспечение, низкая цена: стоимость такого комплекса, как Arduino Uno можно приобрести за две-три тысячи рублей, дополнительные датчики — до двух тысяч рублей [1].

Для приёма данных об изменяющихся условиях необходимо использовать сенсоры [2], [3]. Для водонагревателя, как исполнительного устройства, требуется механизм переключения (включение — выключение). В жёсткой автоматике электромеханическое реле является подходящим устройством для этой цели. Так как температура в помещении расположения объекта обычно немного ниже температуры аквариумной воды, то в имеющейся конфигурации охладитель воды (в виде вентилятора поверхности воды) установлен, но не ис-

пользуется регулярно, а только в критических (сезонных) временных промежутках. Добавление вентилятора необходимо, так как температура в помещении заранее неизвестна.

Алгоритм контроля и управления температурой аквариумного контроллера использует данные о требуемых параметрах среды, в качестве входных: минимальная, максимальная, слишком низкая и слишком высокая температура воды. Приемлемая температура для рыб составляет 24–25°C, на контроллере и зафиксирован этот диапазон входных параметров. Нагреватель регулируется биметаллическим термостатом.

Алгоритм контроллера не обладает свойством самообучения, на этом этапе разумно жёстко закодировать желаемые значения параметров для текущего алгоритма. Для стабилизации температуры используется алгоритм управления — пропорционально-интегрально-дифференцирующий (ПИД) регулятор.

Если температура приемлема, то измеряется потребляемая мощность нагревателя, и работа осуществляется на холостом ходу в течение 15 минут.

Результаты эксперимента показывают, что нагреватель, управляемый внутренним термостатом, невозможно настроить для поддержания стабильной температуры воды с точностью до 1°C. Для демонстрации рабочего поведения нагревателя было проведено сравнение его с реализованным аквариумным контроллером, модуль датчика тока был добавлен для контроля энергопотребления нагревателя.

После сравнения контролирующих температуру воды и других, параметры, которые измеряются до и после того, как регулирование температуры активируется реализованным аквариумным контроллером, следует, что при низкой температуре воздуха в помещении, и при отсутствии фонового регулирования температуры воды, вода нагревается в контуре с фиксированным периодом. Без контроллера цикл нагрева и охлаждения составляет около 4 часов 30 минут. С помощью контроллера, когда диапазон активации нагревателя регулируется до изменения температуры 1°C, средний цикл нагрева и охлаждения воды составляет 3 часа 15 минут.

Таким образом, результаты эксперимента положительные, и может осуществляться дальнейшее совершенствование системы.

Выбранная платформа обладает всеми ожидаемыми ресурсами для достижения большей стабильности параметров объекта, путём совершенствования системы управления аквариумом. Это важный шаг

для повышения точности регулирования параметров воды и отправки предупреждений, когда некоторые параметры выходят за пределы ожидаемого диапазона.

Литература

[1] Сафин И. Р., Краснов А. Н. Актуальность применения бюджетного программно-аппаратного комплекса в учебных заведениях / Сборник трудов IX Международной научно-практической конференции студентов, аспирантов и молодых учёных «Информационные технологии в науке, бизнесе и образовании». Изд.: Московский государственный лингвистический университет (Москва). — 2017 г. — Стр. 79–83.

[2] Сайт, посвящённый разработке программно-аппаратных средств на платформе Arduino. URL: `https://www.arduino.cc/en/Tutorial/LibraryExamples` (дата обращения 14.05.2021).

[3] Сайт платформы Arduino. URL: `https://create.arduino.cc/` (дата обращения 14.05.2021).

Вера Благовещенская, Михаил Черноног
Обнинск, ООО «Базальт СПО»

Тестирование СПО как один из этапов становления ИТ-специалиста

Аннотация

Доклад посвящён обзору направлений в тестировании, особенностям тестирования свободного ПО и перспективам развития в данной области для молодых специалистов. Также мы рассмотрим процесс тестирования заданий, который потом попадают в стабильные репозитории, в компании Базальт СПО.

Считается, что тестирование — это прекрасная возможность попробовать себя в IT. Желающие освоить новую профессию люди, даже без специального образования, могут начать заниматься тестированием. Но не стоит рассматривать тестирование всего лишь как ступеньку к дальнейшей карьере. Тестирование давно уже стало самостоятельной отраслью в мире IT, отдельной технической дисциплиной.

В области, связанной с тестированием, различают понятия QA и QC:

QA (Quality Assurance) — Обеспечение качества

QC (Quality Control) — Контроль качества

QA-инженер помогает **создать** качественный продукт. Он не просто участвует в поиске ошибок. QA-инженер вовлекается в процесс разработки на начальном этапе и тем самым помогает **предотвратить** ошибки.

Люди, которых называют тестировщиками — это QC-инженеры. Они уже на последнем этапе разработки проверяют качество продукта и никак не вовлекаются в процесс создания.

Каждый уважающий себя QC-инженер должен стремиться стать QA-инженером, ведь ошибки легче предотвратить, чем исправлять в уже готовом продукте.

Однако, в России пока не разделяют эти две профессии. Часто для простоты и QA- и QC-инженеров называют тестировщиками.

Хорошие тестировщики влюблены в свою профессию. Такие люди умеют грамотно составлять тестовые сценарии, корректно описывать

найденные ошибки, гибко взаимодействовать с разработчиками. Быть тестировщиком — вовсе не крест на мечте стать программистом. Во-первых, тестирование научит понимать структуру приложений, познакомит с внутренней кухней IT. А во-вторых, есть автоматическое тестирование — очень важная и нужная часть контроля качества продукта.

Нужно ли тестировать свободное ПО? Безусловно — да! Как и любое ПО, свободное ПО также нуждается в тестировании, ведь свободное — не значит бесплатное и плохое.

Конечно, тестирование СПО имеет свои особенности. За годы работы мы накопили некоторый опыт в этом вопросе и постоянно улучшаем процесс контроля качества. Безусловно, залогом успешного тестирования является взаимодействие всех участников проекта и своевременная обратная связь.

В докладе мы рассмотрим такие темы, как:

- мифы о тестировщиках: зачем они нужны и можно ли без них обойтись
- должны ли разработчики тестировать собственный код
- как правильно исследовать и описывать ошибку
- особенности тестирования обновлений для стабильных репозиториев
- обратная связь (общение с разработчиками)
- плюсы и минусы автоматического тестирования
- как начать тестировать свободное ПО

Литература

[1] Джоэл Х. Спольски, «Джоэл о программировании»

[2] Про Тестинг, `https://protesting.ru/testing/`

[3] Кто ты, QA-инженер или тестировщик?, `https://habr.com/ru/company/dododev/blog/497014/`

Денис Ефремов, Алексей Хорошилов
Москва, ИСП РАН

Инструменты тестирования ядра Linux

Аннотация

Обзор инструментов, технологий тестирования и способов организаций тестов, доступных в настоящий момент как в самом ядре Linux, так и в сторонних открытых проектах. Ядро представляет собой большой и неоднородный проект, состоящий как из аппаратно-независимого кода, так и частей, работоспособность которых зависит от наличия соотвествующей аппаратуры или специфичных программ пользовательсткого пространства. Для тестирования разных частей ядра применяются разных методы и подходы. Обзор включает в себя рассмотрение таких инструментов/технологий как kselftest, KUnit, KTF, ktest и наиболее распространённых тестовых пакетов LTP, LKP, xfstests и др.

Модули ядра с тестами

Существует некоторое количество загружаемых модулей ядра, которые выполняют роль тестов. Разработчики пишут эти модули для проверки кода других модулей. Найти их можно, проверив конфигурационные параметры ядра на наличие строк SELFTEST и BENCHMARK в них.

Тесты такого рода не имеют общей организации. Не существует какого-то общего протокола для представления результатов тестирования. Большинство из этих модулей пишут результаты в лог ядра. Лишь некоторые из тестов зависят от наличия специфичной аппаратуры.

Для запуска тестовых модулей необходимо либо собрать ядро вместе с ними, либо собрать их отдельно, в виде загружаемых модулей. В первом случае тесты будут выполняться в момент загрузки ядра, в последнем — в момент инициализации загружаемого модуля.

KUnit

KUnit [2] — это фреймворк для юнит тестирования ядра Linux. В основном он предназначен для небольших тестов, не зависящих от аппаратуры и компонентов пользовательского пространства. KUnit

позволяет организовывать группы тестов, параметризировать тесты, подменять функции (mock), использует TAP [3] протокол для записи результатов тестирования и даёт набор общих программных интерфейсов для разработки тестов. Фреймворк предоставляет общую инфраструктуру для конфигурации, запуска и сбора результатов тестирования. Тесты KUnit могут быть найдены, по наличию строки KUNIT_TEST в конфигурационных параметрах ядра.

KUnit использует UserMode [1] (`ARCH=um`) архитектуру в качестве основной для запуска тестов, но не ограничен ей. Это позволяет запускать ядро Linux как обычный процесс, убирая из тестирования шаг установки ядра и загрузки машины с ним. Тесты запускаются существенно быстрей и работают без использования дополнительного программного обеспечения вроде qemu. Недостатком является то, что на текущий момент UserMode ядро не поддерживает такие отладочные технологиии, как KASAN и UBSAN. KUnit не полагается в своей работе на дополнительное программное обеспечение, не входящее в исходные коды ядра Linux. Фреймворк был включён в ядро Linux v5.5 [4].

KTF

Kernel Test Framework [6] — также является фреймворком для разработки юнит-тестов на ядро Linux. В отличие от KUnit, не входит в само ядро, а является сторонним проектом, развиваемым компанией Oracle.

Фреймворк KTF состоит из ядерной части и программы пользовательского пространства. Последняя контролирует процесс тестирования, запуск тестов, обработку результатов и взаимодействует с компонентом KTF в ядре. Ядерный компонент предостоставляет библиотеку функций для разработки тестов, их запуска и отправку результатов. Результаты представляются в формате TAP. Фреймворк поддерживает параметризацию тестов, их группировку, перехват и подмену функций.

KTF, в отличие от KUnit, требует загрузки тестируемого ядра. Тесты реализуются только в виде загружаемых модулей ядра. Использование UserMode архитектуры при тестировании не поддерживается.

С другой стороны, KTF использует сокеты netlink (а не лог ядра) для записи результатов тестирования и контроля запуска тестов, а также использует технологии Kprobes/Kretprobes для перехвата

внутренних API ядра. Всё это позволяет реализовывать гибридные тесты, которые выполняются и в пространстве пользователя и в пространстве ядра, что даёт возможность тестировать как внешнее воздействие на ядро отражается на его внутреннем состоянии. Полезными особенностями фреймворка является автоматизация доступа к внутренним неэкспортируемым API ядра и поддержка упрощённого покрытия кода, не требующего пересборки ядра.

К сожалению, в настоящий момент нет репрезентативного и публично доступного набора тестов, основанного на фреймворке KTF. После ухода из Oracle основного разработчика KTF проект поддерживается, но не продолжает своего развития. Была лишь одна попытка включить KTF в ядро Linux, которая имела ряд замечаний, в том числе замечание о необходимости его интеграции с KUnit. Дальнейших попыток включить KTF в ядро на сегодняшний день не предвидится.

Kselftest

Linux Kernel Selftests [5] — это набор тестовых программ пользовательского пространства для интеграционного и сквозного тестирования, который поставляется вместе с исходниками ядра Linux. Данные тесты предназначены для запуска после сборки, установки и загрузки тестируемого ядра.

Отдельный тест представляет собой пользовательскую программу, которая взаимодействует с ядром посредством системных вызовов, возможно также полагаясь на наличие в системе специфичных интерфейсов `/dev/*, /proc/*, /sys/*` отдельных драйверов.

Kselftest использует протокол TAP для записи результатов тестирования и поддерживает множество компьютерных архитектур. Тесты осуществляют проверку того, что API для программ пользовательского пространства функционируют ожидаемым образом. Основной целью проекта является помочь разработчикам ядра производить базовое тестирование (включая регрессионное) менее чем за 20 минут.

Ktest

Более 10 лет в составе ядра существует скрипт ktest.pl [7], предназначенный для автоматизации тестирования ядра. Скрипт поддерживает как работу с физическими оборудованием, так и виртуальные машины. Ktest позволяет производить тестирование сборки и загруз-

ки ядра, в том числе с выполнением некоторого набора тестов после загрузки. Ktest имеет режимы проверки отдельных патчей, поиска коммитов (git bisect) с ошибками, минимизации конфигурации ядра.

В рамках тестирования сборки и загрузки ядра, ktest может сверять результаты с прошлыми запусками и выявляеть новые ошибки/аномалии на основании этого.

Ktest в основном разрабатывался как инструмент мейнтейнеров ядра Linux для ревью и тестирования патчей из списков рассылок. Соответственно, он обладает функциональностью необходимой для максимальной автоматизации и упрощения данного рабочего процесса.

Тестовые пакеты

Существует большое количество тестовых пакетов, нацеленных на разные подсистемы и функциональность ядра Linux. Данные тесты поддерживаются и разрабатываются разными мейнтейнерами и компаниями. Это программы пользовательского пространства, нацеленные на регрессионное, функциональное, интеграционное, сквозное, стресс тестирование интерфейсов ядра или системы в целом, её производительности. Они не имеют какой-то общей структуры, представления результатов, и в отдельных случаях не имеют оракулов для отслеживания ошибок ядра во время тестирования. В последнее время такие continuous integration проекты как CKI, LKFT, KernelCI, 0-Day Test Service начали активно использовать данные тестовые пакеты и включать их в цикл непрерыной интеграции.

Вот только некоторые из публично доступных тестовых пакетов на ядро Linux:

- Linux Test Project (LTP) [8] — большой тестовый пакет на практически все подсистемы ядра, включая стресс тесты и тесты на наличие неисправленных уязвимостей. Развивается и поддерживается крупными компаниями.

- Linux Kernel Performance tests (LKP) [9] — тесты производительности подсистем виртуальной памяти, ввода-вывода, сетевой и др. Развивается Intel, используется в O-Day Continuous Integration системе с автоматической отправкой результатов в публичные списки рассылки.

- xfstests [10] — набор регрессионных тестов для операций с файловыми системами. Изначально разрабатывался для тестирования XFS, но постепенно стал основным инструментом тестирования для всех файловых систем ядра Linux.
- syzkaller-repros [11] — коллекция программ вызывающих ошибки в ядре, найденных при помощи системы фаззинга системных вызовов syzkaller. Используется в качестве набора регрессионных тестов.
- Firmware Test Suite [12] — тесты для проверки прошивок. Нацелен на выявление ошибок BIOS, UEFI, ACPI и др.
- KVM unit tests [13]/Xen Test Suite [14] — тесты для проверки гипервизоров KVM и XEN. Представляют собой наборы минимальных виртуальных машин разных конфигураций. В тестовый набор XEN включаются регрессионные тесты на выявления известных уязвимостей XSA.

Остальные тестовые пакеты на ядро Linux практически наверняка можно найти в базе KCIDB [15] системы непрерывной интеграции KernelCI.

Литература

[1] UML HowTo `https://www.kernel.org/doc/html/latest/virt/uml/user_mode_linux_howto_v2.html`

[2] KUnit - Unit Testing for the Linux Kernel `https://www.kernel.org/doc/html/latest/dev-tools/kunit/index.html`

[3] Test Anything Protocol `https://testanything.org/`

[4] kunit: test: add KUnit test runner core `https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/commit/?id=914cc63eea6f`

[5] Linux Kernel Selftests `https://www.kernel.org/doc/html/latest/dev-tools/kselftest.html`

[6] Kernel Test Framework `https://github.com/oracle/ktf/`

[7] Ktest `https://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git/tree/tools/testing/ktest`

[8] Linux Test Project `https://github.com/linux-test-project/ltp`

[9] Linux Kernel Performance tests `https://github.com/intel/lkp-tests`

[10] (X)FSTests filesystem testing suite https://git.kernel.org/pub/scm/fs/xfs/xfstests-dev.git/

[11] Syzkaller bug reproducers https://github.com/dvyukov/syzkaller-repros

[12] Firmware Test Suite https://wiki.ubuntu.com/FirmwareTestSuite

[13] KVM unit tests https://www.linux-kvm.org/page/KVM-unit-tests

[14] Xen Test Framework http://xenbits.xenproject.org/docs/xtf/

[15] KCIDB https://github.com/kernelci/kcidb/blob/2da9805978e73b2bbba3c1d5996c0d49d7944163/tests.yaml

Стас Фомин, Кирилл Герасимов
Москва, ИСПРАН

Проект: TerrariumAssembler http://wiki.4intra.net/TerrariumAssembler, http://github.com/belonesox/terrarium-assembler

Terrarium Assembler — эффективные сборка и распространение высокотехнологичных Python-приложений под различные операционные системы

Аннотация

Сейчас Python — самый удобный язык, по крайней мере для прототипирования и разработки высокотехнологических приложений, ибо сейчас в нём есть «батарейки» практически для всех областей высоконаучной разработки — оптимизация, машинное обучение, симуляции, распознавание текста (sympy, numpy, cvxopt, or-tools, opencv, torch, keras, catboost, xgboost, h2o. . .).

Однако, перейти от прототипа, к собранному приложению, которое можно выкатить на зоопарк разных, местами очень древних версий Windows и Linux, включая «замученные и урезанные» сертифицированные российские линуксы, весьма непросто. При этом обычно нужно решить проблему защиты алгоритмов и защиты от реверсинжиниринга.

Столкнувшись с такой проблемой, мы разработали серию open-source проектов «Terrarium Assembler», решающую эту задачу.

К 2020 году, исполнилось 30 лет языку Python, и при этом он стал самым популярным языком в рейтинге TIOBE. За это время, Python, идеальный язык для обучения, рождённый как язык для обучения от уровня младших школьников, стал использоваться профессионалами во всех областях, как в программировании, так и в куче научных областей — будь то астрономия, биоинформатика, статистика... везде.

Можно долго рассуждать о причинах такого успеха — повлиял ли простой синтаксис, понимаемый не только профессиональными разработчиками, но и любым неглупым пользователем, «дзен питона», ориентированный не на удобство компьютера, а пользователя, максимальная простота и читаемость, удобство для пользователя в разработке и отладке, выбирались при любых инженерных дилеммах построения языка. Но это неважно, а важно то, что сейчас Python — самый удобный язык, по крайней мере для прототипирования и разработки высокотехнологических приложений, ибо сейчас в нём есть «батарейки» практически для всех областей научной разработки:

- Криптография
- Символическая алгебра — sympy
- Линейная алгебра — numpy
- Алгоритмы оптимизации — cvxopt, or-tools
- Машинное зрение — opencv
- Artificial Intelligence и Machine Learning — torch, keras, catboost, xgboost, h2o...

Если раньше, для разработки и прототипирования новых алгоритмов нужно было прибегать к использованию специализированных «пакетов для математиков» — систем типа Mathematica, Mathcad, Mapple и т.п., с переписыванием их для реального использования на языки промышленной разработки (C/C++/Java), то сейчас, разработку новых алгоритмов принято вести на Python, причём заменяя или дополняя привычные «статьи с псевдокодом», алгоритмы в которых невозможно проверить и исследовать без реализации, на Jupyter-ноутбуки, являющиеся гибридами чередующегося текста, формул, и работающего и проверяемого кода на Python, выдающего верифицируемые графики и визуализации.

Более того, движение `https://paperswithcode.com/`, и множество статей практически констатирует текущий кризис воспроизводимости

научных результатов, призывает, чтобы все научные результаты были в таком формате.

Причём разработку новых алгоритмов на Python можно вести на всех этапах — от первых идей, через бурю множества итераций, до выкатки в реальное использование, таким образом, получая обратную связь по тестированию непосредственно от пользователей, и итерационно улучшая алгоритмы без долгих циклов переписывания на другие языки и фреймворки. И лишь в крайних случаев, для небольших и критических кусков кода, прибегая к ускорению и реализации в виде нативных C-модулей.

Да, сам интерпретатор Python существует практически под все платформы и операционные системы, и потенциально, многие вообще не видят проблем — ну вот, код на Python — «почему бы ему не работать на всём, от привычных x86 до ARMов, MIPSов и Эльбрусов, от древних версий Windows, до сертифицированных российских линуксов?».

К сожалению, с этим всё непросто:

- Базовый интерпретатор Python, как правило да, есть везде.
- Но вот сборка всех необходимых стеков Python-модулей — серьёзная проблема, по крайней мере, для старых версий Windows и малораспространённых линукс-дистрибутивов (к которым относятся и российские сертифицированные линукс-дистрибутивы), пакетная база которых в десятки раз меньше, чем у популярных дистрибутивов с большим сообществом майнтейнеров (Fedora-Debian-Ubunta), а используемые базовые библиотеки серьёзно устарели, так, что собрать современный прикладной стек поверх них попросту невозможно.
- Попытка сделать «единый дистрибутив пакет», с минимумом вариаций (windows/linux), не сильно менее проста, особенно в связи с слабой обратной бинарной совместимостью (libc, ld.so) линуксов, и отсутствием поддержки в старых сертифицированных линуксах технологий изоляции и контейнеризации (docker/flatpack/snapd), которые могли бы эту проблему решить.

Отдельно, стоит проблема сохранения интеллектуальной собственности. Конечно, open-source и открытость научных результатов — это прекрасно. «Если у вас есть идея, и у нас есть идея — то обменявшись — у нас будет две идеи» — говорят сторонники открытости. Однако,

это работает не всегда — если мы обменяемся секретами, то секреты просто исчезнут.

В ряде случаев, особенно, когда речь идёт про информационную безопасность, несмотря на то, что «Security by Obscurity» сам по себе не является надёжным принципом для ИБ, это может создавать дополнительный уровень защиты — при условии, разумеется, применения всех остальных принципов надёжной и верифицируемой разработки. Не говоря уже, о конкурентном преимуществе вашего решения, которое может быть даже в наборе некоторых констант и небольших улучшений к известным алгоритмам, но найденных вами в результате тяжёлых лет исследований.

Вот тут, увы, у Python всё было не очень хорошо. Да, существовали методы получения скомпилированных утилит из Python кода, таких как Py2exe, PyInstaller, и т.п., но все они по сути, «зашивали» в выполняемый файл готовый интерпретатор питона, и байт-код скомпилированных python-библиотек. Декомпилировать всё это, несмотря на некоторые возможные методы обфускации — задача увы, технически простая (см. decompile6).

И так получилось, что в ходе одного из проектов ИСПРАН, возникла необходимость разработки решения, в области борьбы с утечками данных, для которого:

- Не существовало хороших известных решений.
- Разрабатываемые новые алгоритмы активно используют алгоритмы компьютерного зрения, распознавания текста, AI/ML и т.п. — по вышеуказанным причинам, никакой другой стек (C++, Go, Rust...) не позволил бы эффективную разработку таких алгоритмов непосредственно с обкаткой и сбором обратной связи от заказчика.
- Решение должно выкатываться на сотни тысяч компьютеров, и множество операционных систем, как правило различные 32-битные и 64-битные версии Windows, и 64-битные сертифицированные линуксы, разных версий и вендоров (Astra Linux, ALT Linux) и т.п, содержало клиентскую и серверную часть.
- Код-алгоритмы-константы должно быть нельзя восстановить-реинженирить, используя установленную систему.
- Но при этом, решение должно пройти сертификацию в специализированной лаборатории, и поэтому, должна существовать

простая и автоматизированная система сборки, где все артефакты — код, библиотеки, можно было бы проследить, проанализировать на уязвимости и недокументированные возможности.

  – Должна быть реализована «сборка в чистой комнате», без доступа в интернет, и скачивания чего бы то ни было — только такой вариант даёт хоть некоторую гарантию воспроизводимости сборки. Для воспроизводимости, сборка должна «бутстрапится» и выполняться с использованием простых командных файлов.

- При этом, система сборки должна быть достаточно удобна и гибка для разработчиков, чтобы можно было легко экспериментировать с различными версиями библиотек.

Для решения этих задач, был реализован комплекс управления сборкой Terrarium Assembler, с версиями под Linux и Windows, и другие вспомогательные проекты, которые были опубликованы в *open-source*.

В докладе, мы рассмотрим вышеупомянутые проблемы, технологические дилеммы, и наши решения более подробно.

Андрей Русалин, Сергей Козьяков
Москва, Иннополис, ООО «Открытая Мобильная Платформа»

Проект: ОС «Аврора» `https://auroraos.ru/`

Открытые смартфоны: прошлое, настоящее, будущее

Аннотация

Современные мобильные устройства обременены большим количеством закрытого кода и компонентов, что делает разработку нового системного программного обеспечения для существующих устройств сложным, длительным, дорогостоящим процессом. На протяжении недолгой истории смартфонов встречалось несколько устройств, которые можно было бы назвать «открытыми», но данные проекты не снискали популярности и крупного коммерческого успеха. На данный момент выделяется три устройства — Necunos, Librem 5 и PinePhone. Данный доклад рассматривает прошлые и текущие открытые устройства, аппаратные и программные компоненты, используемые в них; обсуждаются сложности разработки и адаптации программного обеспечения, и оцениваются перспективы и будущее подобных проектов.

Проблемы разработки ПО под закрытые платформы

Современные мобильные устройства (МУ) обременены большим количеством закрытых программных и аппаратных компонентов. «Закрытые», или «проприетарные» компоненты здесь противопоставляются «открытым», т.е. компонентам, распространяемым под открытой лицензией и с открытым исходным кодом. Закрытые компоненты поставляются в виде бинарных программ, библиотек, драйверов — так называемых «блобов». Так, при адаптации новой ОС (например, «ОС Аврора») для устройств с уже существующим ПО от поставщика, количество блобов, необходимых для работы системы, может исчисляться десятками. Часто такие компоненты не снабжены документацией и сторонние разработчики не могут использовать полный их функционал. В случае возникновения проблем не обойтись без взаимодействия с производителями чипов, что не всегда возможно по разным причинам. Производитель может не взаимодействовать с сообществом вообще или не взаимодествовать по определённым моделям чипов, а если речь идёт про конкретные коммерческие проекты, то уровень поддержки часто зависит от объёма проекта.

Однако в последнее время возрос интерес к открытому коду и устройствам. Производители нередко публикуют исходный код драйверов и библиотек для своих продуктов и принимают исправления и нововведения от сообщества. Историю появления открытых компонентов в МУ, саму практику использования таких компонентов и причины интереса к данной практике мы и рассмотрим в данном докладе.

Прошлое открытых МУ

GreenPhone, выпущенный в 2006 году компанией Trolltech, был одним из первых МУ с ОС на базе ядра Linux и открытыми компонентами. Главным фреймворком и средой устройства была Qtopia, доступная под лицензией GPL [1].

В том же 2006 году был представлен проект Openmoko, направленный на создание открытых МУ, аппаратных спецификаций и операционной системы (Openmoko Linux). В 2007 году было представлено устройство Neo 1973, для которого, помимо открытого ПО, под открытой лицензией доступна схематика и чертежи.

Это не единственные устройства, позиционируемые как открытые смартфоны, однако большинство из других примеров либо были предназначены только для разработчиков и не выпускались на рынок, либо не являются полностью открытыми устройствами.

Современные открытые МУ

Рассмотрим три интересных проекта открытых МУ, представленных на рынке за последние несколько лет.

Во-первых, это устройства Necunos, позиционируемые как «безопасные и проверяемые МУ[2]». Они не оснащены модемом, поэтому не могут совершать звонки и использовать LTE соединение. В итоге, по заверению разработчиков, устройство поставляется с одним единственным«блобом», необходимым для работы Wi-Fi чипа. Предполагается, что устройства будут поставляться с Aera OS, основанной на ядре Linux. В настоящий момент исходные коды и BSP недоступны, но, по заверениям разработчиков, будут опубликованы «в ближайшем времени».

Во-вторых, это Librem 5 от компании Purism, производящей различные устройства, такие как ноутбуки и мини-ПК, ориентированные на конфиденциальность пользователя. Устройство поставляется с оболочкой Phosh, основанной на GNOME, и переиспользует много существующего ПО. Например, для звонков используется фреймворк oFono, созданный в своё время компанией Intel, а ныне поддерживаемый сообществом, и используемый в том числе в ОС Аврора и других мобильных ОС на основе ядра Linux.

Наконец, особый интерес для нас представляет PinePhone от компании Pine64, которая до этого занималась выпуском одноплатных компьютеров и тесно взаимодействует с сообществами открытых проектов. Интересно это устройство тем, что с ранних дней были доступны наборы разработчиков и прототипы устройства. Дистрибутивы, множество которых сейчас доступно для загрузки и тестирования[3], также активно используют и развивают различные программные и низкоуровневые компоненты.

Открытое ПО и взаимодействие с сообществом

Можно проследить, что ПО, используемое в ОС на базе Linux современных МУ, в том числе ОС «Аврора», пересекается с проектами,

активно используемыми как встраиваемыми системами, так дистрибутивами для персональных компьютеров. Это приводит к упрощению и снижению стоимости разработки, а активное участие сообщества в разработке помогает адаптировать эти компоненты для огромного множества МУ. Рассмотрим некоторые программные компоненты, используемые в устройствах, как в вышеупомянутых открытых, так и в «закрытых».

Уязвимости, затрагивающие одно устройство, по своей натуре часто могут затрагивать множество других, но и исправление уязвимостей в открытых проектах становится доступно для прочих устройств. Так, в 2020 году разработчики ОС «Аврора» обнаружили и исправили критическую проблему в glibc, затрагивающую множество устройств на архитектуре ARMv7 [4], и это исправление теперь доступно для всего сообщества. Подобное взаимодействие, где важное исправление может быть быстро опубликовано и доступно для разработчиков по всему миру, очень важно для современной разработки, и поэтому даже крупные разработчики закрытых компонентов могут быть заинтересованы в использовании открытых проектов в своей разработке и отправки изменений в них.

В отличие от более старых мобильных дистрибутивов, вместо проекта Xorg для графики используются реализации Wayland. Пользовательское окружение может быть простроено как на фреймворке Qt, так и на GTK. Для мультимедиа часто используется проект GStreamer.

Для управления звуком нередко используется PulseAudio, а проект Mer предлагает набор расширений для интеграции звуковой подсистемы с остальной ОС, а также утилиты для управления политиками и звуковыми профилями устройства. При разработке ОС «Аврора» эти расширения активно используются и дорабатываются, а изменения отправляются обратно в сообщество.

Благодаря своей модульной структуре, oFono с расширениями из проекта Mer может использовать компоненты ОС Андройд для совершения вызовов через проприетарные компоненты или использовать открытые менеджеры различных устройств. Так, разработчики Librem 5 реализовали поддержку модема, используемого в этом МУ. Наработки были опубликованы и частично были переиспользованы в поддержки других модемов, в т.ч. в PinePhone.

Это не все компоненты, наработки в которых могут переиспользоваться. При разработке ОС «Аврора» часто добавляется и оптимизи-

руется поддержка различного функционала, связанного с голосовыми вызовами, которая апстримится в проект Mer.

Будущее открытых смартфонов

«Открытая Мобильная Платформа» следит за развитием современных открытых мобильных устройств и принимает активное участие в разработке программных компонентов. Хотя сейчас рано говорить о коммерческом успехе современных открытых МУ, особенно учитывая их высокую стоимость и местами неидеальное ПО, популяризация разработки и использования открытого ПО и запрос на прозрачность дают основания полагать, что популярность подобных устройств будет расти.

Литература

[1] linuxdevices.com — Trolltech GPL's Qtopia app stack for Linux PDAs, https://web.archive.org/web/20090105054754/http://www.linuxdevices.com/news/NS9985953607.html

[2] Necunos — Necunos FAQ, https://necunos.com/faq/

[3] Pine64 — https://wiki.pine64.org/wiki/PinePhone_Software_Releases

[4] CNews — Разработчики ОС «Аврора» исправили уязвимость в glibc, https://mobile.cnews.ru/news/top/2020-07-16_razrabotchiki_os_avrora

Бондарев Антон Владимирович
Санкт-Петербург

Проект: Embox http://embox.github.io/

Embox — свободная ОСРВ позволяющая запускать сложные C++ приложения на микроконтроллерах

Аннотация

Доклад посвящён особенностям работы C++ приложений на микроконтроллерах. В частности рассматриваются причины ограничений C++ по сравнению с десктопными средствами разработки на данном языке. Рассматриваются особенности реализвации поддерки C++ в

ОСРВ Embox. Поскольку данныа ОСРВ позволяет избежать ограничений и имеет наиболее полную поддержку C++ для микроконтроллеров.

Сегодня существует множество средств предоставляющих средства разработки пользовательских приложений на C++ для микроконтроллеров. Однако большинство из них содержать некоторые ограничения. Прежде всего это невозможность перезапуска приложений и невозможноть использования функций из стандарта C++ связанных с потоками.

Embox — открытая операционная система реального времени [1], позволяет избежать этих ограничений, то есть использовать подобную функциональность на микроконтроллерах. Данная статья посвящена деталям реализации рантайма C++ в ОСРВ Embox.

Базовый синтаксис

Синтаксис языка C++ реализуется компилятором. В рантайм необходимо реализовать несколько базовых сущностей. В компиляторе они включаются в библиотеку поддержки языка libsupc++.a. Наиболее базовой является поддержка конструкторов и деструкторов. Существуют два типа объектов: глобальные и выделяемые с помощью операторов new.

Рассмотрим то как работает C++ приложение. Перед тем как попасть в main(), создаются все глобальные C++ объекты которые присутствуют в коде. Для этого используется специальная секция .init_array. Также могут использоваться секции .init, .preinit_array, .ctors. Для современных компиляторов ARM, чаще всего секции используются в следующем порядке .preinit_array, .init и .init_array. С точки зрения LIBC это обычный массив указателей на функции, который нужно пройти от начала и до конца, вызвав соответствующий элемент массива. После этой процедуры управление передаётся в main().

Рассмотрим как устроено завершение C++ приложения, а именно, вызов деструкторов глобальных объектов. Существует два способа.

Наиболее используемый в компиляторах — через __cxa_atexit() (из C++ ABI). С помощью этой функции могут быть зарегистрировать специальные обработчики, которые будут вызваны в момент завершения программы. Когда при старте приложения происходит вызов глобальных конструкторов, как описано выше, кроме того ту-

да включается сгенерированный компилятором код, который регистрирует обработчики через вызов __cxa_atexit. Задача LIBC здесь сохранить требуемые обработчики и их аргументы и вызвать их в момент завершения приложения.

Другим способом является сохранение указателей на деструкторы в специальных секциях .fini_array и .fini. В компиляторе GCC это может быть достигнуто с помощью флага -fno-use-cxa-atexit. В этом случае во время завершения приложения деструкторы должны быть вызваны в обратном порядке (от старшего адреса к младшему). Этот способ менее распространён, но может быть полезен в микроконтроллерах. Ведь в этом случае на момент сборки приложения можно узнать сколько обработчиков потребуется.

Глобальные деструкторы необходимы, чтобы иметь возможность перезапускать C++ приложения. Большинство RTOS для микроконтроллеров предполагает запуск единственного приложения, которое не перезагружается. Старт начинается с пользовательской функции main, единственной в системе. Поэтому в небольших RTOS зачастую глобальные деструкторы пустые.

В компиляторе GCC реализация операторов new/delete находится в библиотеке libsupc++.

Можно использовать реализации new/delete из libsupc++.a, но они достаточно простые и могут быть реализованы например, через стандартные malloc/free или аналоги.

Если приложение простое, вам не требуется поддержка исключений и динамическая идентификация типов данных (RTTI). Её можно отключить с помощью флагов компилятора -no-exception -no-rtti.

Но если эта функциональность C++ требуется, её нужно реализовать. Сделать это куда сложнее чем new/delete.

Но так как эта функциональность на зависит от ОС можно воспользоваться реализацией библиотеки ibsupc++.a из кросс-компилятора.

Для использования исключений из кросс-компилятора есть небольшие требования, которые нужно реализовать при добавлении собственного метода загрузки C++ рантайма. В линкер скрипте должна

быть предусмотрена специальная секция .eh_frame. А перед использованием рантайма эта секция должна быть инициализирована с указанием адреса начала секции.

Для ARM архитектуры используются другие секции с собственной структурой информации — .ARM.exidx и .ARM.extab. Формат этих секций определяется в стандарте «Exception Handling ABI for the ARM Architecture» — EHABI[2]. .ARM.exidx это таблица индексов, а .ARM.extab это таблица самих элементов требуемых для обработки исключения. Чтобы использовать эти секции для обработки исключений, необходимо включить их в линкер скрипт.

Чтобы GCC мог использовать эти секции для обработки исключений, указывается начало и конец секции .ARM.exidx — __exidx_start и __exidx_end. Эти символы импортируются в libgcc в файле libgcc/unwind-arm-common.inc:

```
extern __EIT_entry __exidx_start;
extern __EIT_entry __exidx_end;
```

Стандартная библиотека языка (libstdc++)

В поддержку языка C++ входит не только базовый синтаксис, но и стандартная библиотека языка libstdc++. Её функциональность, так же как и для синтаксиса, можно разделить на разные уровни. Есть базовые вещи типа работы со строками или C++ обёртка setjmp . Они легко реализуются через стандартную библиотеку языка C. А есть более продвинутые вещи, например, Standard Template Library (STL).

При использовании стандартной библиотеки C++ из кросс-компилятора существует особенность. Взглянем на стандартный arm-none-eabi-gcc. Он собран с поддержкой —with-newlib. Newlib реализация стандартной библиотеки языка C. В Embox используется собственная реализация стандартной библиотеки. Так как стандартные библиотеки C отличаются, то для поддержки рантайма нужно реализовать слой совместимости.

Кроме стандартной библиотеки кросс-компилятор имеет модель потоков “Thread model: single”. Когда GCC собран с этой опцией, убирается вся поддержка потоков из STL (например std::thread и std::mutex). И, например, со сборкой такого сложного C++ приложение как OpenCV возникнут проблемы. Иначе говоря, для сборки

приложений, которые требуют подобную функциональность, недостаточно этой версии библиотеки.

Решением, которые мы применяем в Embox, является сборка собственного компилятора ради стандартной библиотеки с многопоточной моделью. В случае Embox модель потоков используется posix “Thread model: posix”. В этом случае std::thread и std::mutex реализуются через стандартные pthread_* и pthread_mutex_*. При этом также отпадает необходимость подключать слой совместимости с newlib.

Денис Силаков
Москва, Virtuozzo

Проект: OpenVZ `https://openvz.org/`

Virtuozzo Linux 8 и OpenVZ 8 — текущее состояние и планы

Аннотация

С целью упрощения поддержки, OpenVZ 7 изначально основывался на ядре и пакетной базе Red Hat Enterprise Linux (RHEL). В 2019 году вышел RHEL 8, и уже более двух лет идут работы по переводу ядра и утилит OpenVZ на эту версию. В качестве первого шага, был налажен процесс сборки и обновления дистрибутива Virtuozzo Linux 8. В силу планов Red Hat по переводу CentOS 8 на «streaming» модель, пользователи Virtuozzo/OpenVZ высказали заинтересованность в более стабильных альтернативах, поэтому теперь VzLinux 8 позиционируется и как отдельный продукт, уже готовый к использованию. Данный доклад рассматривает как технологические процессы разработки VzLinux 8, так и текущее состояние и перспективы OpenVZ 8.

Заметная задержка между выходом RHEL 8 и Vz/OpenVZ 8 вызвана традиционной неспешностью клиентов Virtuozzo — в момент выхода RHEL 8, многие крупные пользователи только начали переход на Vz 7 и заметная доля усилий разработчиков была направлена на удовлетворение их потребностей. Только спустя год, удалось сместить акцент на разработку новой версии.

VzLinux 8

Пакетная база в виде Virtuozzo Linux 8 была готова вскоре после выхода RHEL 8. Выпускать этот дистрибутив как отдельный продукт изначально не планировалось, однако обеспокоенность пользователей судьбой CentOS 8 изменила планы и теперь мы предлагаем VzLinux 8 как гостевую ОС в контейнерах и ВМ на замену CentOS 8. Это создаёт дополнительную нагрузку на разработчиков (например, изначально не планировалось иметь в VzLinux всё многообразие модулей и стримов из CentOS 8), однако позволяет иметь более предсказуемого гостя в контейнере.

В этого гостя можно вносить изменения, предназначенные именно для работы в контейнерах (например, предоставлять как nftables, так и legacy-инструменты iptables, демонстрирующие в ряде сценариев лучшую производительность), и исправлять проблемы оперативно, не дожидаясь обновлений от Red Hat (по схожим причинам, в основе самого OpenVZ начиная с седьмой версии лежит не сам CentOS, а его форк VzLinux).

OpenVZ 8

В OpenVZ 8 предполагается обеспечить обратную совместимость с OpenVZ 7 — даже если какие-то технологии претерпят радикальные изменения, будет предоставлен слой совместимости для запуска виртуальных окружений, мигрировавших с предыдущей версии продукта, либо инстурментарий для их преобразования при миграции. То же самое относится к инструментальным средствам — все стандартные утилиты OpenVZ 7 будут доступны и в новой версии, хотя ряд из них будет объявлен устаревшим и не будет устанавливаться по умолчанию.

Из инструментов, наиболее заметным изменением является перевод prlctl и PRL SDK в разряд устаревших. Соответствующее сообщение уже отображается в последнем релизе OpenVZ 7 при использовании prlctl и сопутствующих утилит. Для работы с виртуальными машинами по умолчанию будет предлагаться virsh и libvirt API, которые предполагается доработать для большей совместимости с prlctl/PRL SDK. Часть функционала prlctl и prlsrvctl уйдёт в другие инструменты (например, ручная отсылка отчётов об ошибках полностью перейдёт в vzreport, который и занимался этим до появления

prlctl). Для работы с контейнерами, libvirt в сборке OpenVZ давно содержит vzct драйвер. Впрочем, убирать vzctl (в отличие от prlctl) пока не планируется.

Как в инструментарии, так и в ядре, планируется продолжить по возможности переносить изменения Vz в апстрим и переиспользовать другие появляющиеся в апстриме наработки. Например, при работе с траффиком контейнеров избавиться от модуля ядра vznetstat в пользу nft, переписать собственный канал общения хоста с гостевым агентом QEMU на использование viosock, использовать ipvlan вместо venet/veth и так далее.

Отдельное направление работ планируется в сфере хранения данных контейнеров. В рамках унификации управления контейнерами и ВМ через libvirt, возможно добавление storage-драйвера ploop — чтобы «вписать» ploop в концепцию Storage Pools/Volumes. В то же время рассматривается вариант использование для контейнеров формата QCOW2, как для ВМ. Одной из целей добавления ploop в коммерческой версии Virtuozzo был единый формат дисков для ВМ и контейнеров, однако в седьмой версии произошёл переход на QEMU/KVM и теперь этот аргумент неактуален.

Расписание релизов

Альфа-версию OpenVZ планируется представить уже этим летом. Релиз планируется в первом полугодии 2022 года. За разработкой открытой части можно следить в гите OpenVZ (`https://src.openvz.org/`) в соответствующих бранчах ядра и инструментов.

Костарев Алексей Фёдорович
Пермь, ООО «Новая Платформа»

Проект: Репозиторий Sisyphus `http://www.sisyphus.ru/`

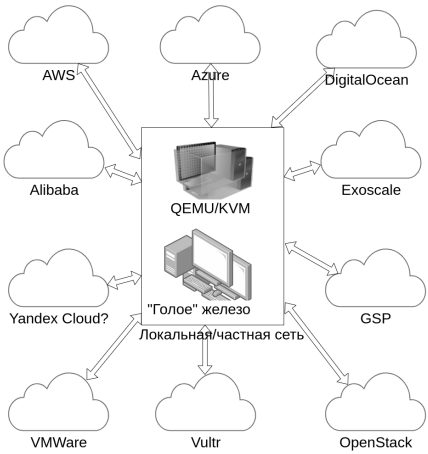
Fedora CoreOS. Мы Федоре не враги

Аннотация

В докладе будут рассмотрены история возникновения и ключевые особенности дистрибутива «Fedora CoreOS» компании RedHat и его аналогов. Возможность реализации основных идей дистрибутива в рамках операционной системы ALT Linux.

Как правило, в различных средах разворачиваются различные Linux-дистрибутивы с различными версиями операционных систем, системных библиотек и т.п.. В рамках каждой версии на различных узлах устанавливается различное программное обеспечение (ПО). Если число узлов в кластере относительно невелико (несколько узлов) администрирование операционных систем (установка обновлений, закрытие уязвимостей, ...) на узлах является небольшой проблемой. Если же число узлов исчисляется десятками и сотнями, грамотное администрирование разнородных операционных систем с различных установленным ПО становится невозможным.

Fedora CoreOS предлагает решение данной проблемы за счёт использования следующих механизмов:



- Исключения участие оператора при разворачивании операционной системы. Для каждого узла в файле конфигурации задаются параметры развёртывания (разбиение дисковой подсистемы, конфигурирование сетевых интерфейсов и DNS, настройка сервисов и контейнеров, ...). Установка операционной системы на различные вышеперечисленные платформы производится без участия оператора.
- Автоматическое обновления ядра и операционной системы из одного центра без участия оператора с возможностью приоста-

новить автоматическое обновление и откатить на предыдущую ветку.

- Использование OSTree (git для бинарных файлов) позволяет:
  - откатывать состояние системы на предыдущие коммиты (точки фиксации)
  - разделить файловую систему на неизменяемую (`/usr/`), изменяемую (`/var/`) и обновляемую (`/etc`) части. Что позволяет:
    - упростить обновление бинарного программного кода системы;
    - исключить случайное удаление программного кода;
  - перегружать систему на предыдущие или последующие коммиты.
  - в случае необходимости администратор может, используя утилиту rpm-ostree доустановить или удалить пакеты, перебазироваться на другое дерево, заменить параметры ядра и т.д.
- Администратор кластера может останавливать часть узлов кластера при снижении нагрузки и восстанавливать их при повышении.
- Весь пользовательский софт работает в контейнерах (podman, moby engine == docker) в различных средах контейнеризации (docker swarm. k8s, ...). Это упрощает обновление системы и исключает изменение функционала пользовательского ПО после обновления системы.
- Файловая система Fedora CoreOS достаточно небольшая (менее 4GB), что позволяет её использовать в устройствах Интернета Вещей.

Содержание операционной системы:

- последние свежие базовые компоненты Fedora (RPM-пакеты);
- максимальная поддержка разнообразного железа (драйверы);
- Основные (терминальные) административные инструменты;
- Контейнерные движки: podman, moby engine.

Потоки разработки Fedora CoreOS

Релизы Fedora CoreOS выпускаются в трёх параллельных потоках:

- next — реализация экперементального функционала. После стабилизации выпускаются мажоные версии дистрибутива.
- testing — стабилизация RPM-пакетов. После стабилизации переводится в поток stable;
- stable — самый надёжный поток из перечисленных (публикация каждый 2 недели).

Пользователи могут подключаться к любому из этих потоков и принимать участие в тестировании ближайщего или эксперементального функционала или использовать поток stable для повседневной работы.

Fedora CoreOS vs Fedora IoT

Наряду с дистрибутивом Fedora CoreOS компания RedHat продвигает близкое по концепции решение для Интернета Вещей: Fedora IoT.

Сходства дистрибутивов:

- Использование ostree, rpm-ostree для поддержания неизменяемой операционной системы с атомарными изменениями.
- Использование igniion для автоматического разворачивания на множество аппаратных устройств.
- Пользовательские приложения разворачиваются в контейнерной среде.
- Поддержка TPM2, SecureBoot и автоматического расшифровки данных с устройств хранения с помощью Clevis.

Различия дистрибутивов:

- поддержка различных аппаратных платформ: x86_64, aarch64, armhfp.
- упор на поддержку интернет-устройств и драйвером к ним;
- меньший объём дискового прострнства ~1.5GB
- контейнеризация осуществляется через podman, поддержка клиент-серверной модели moby engine (docker) отсутствует;
- веб-служба Zezere с помощью которого администраторы могут развёртывать и настраивать Fedora IoT масштабируемым образом без физической консоли.

Альт CoreOS

В настоящее время на основе репозитория Sisyphus «Базальт СПО» выпускает следующую линейку дистриутивов: *Альт Рабочая станция*, *Альт Сервер*, *Альт Сервер Виртуализации*, *Альт Образование*, *Simply Linux*.

Использую технологии и программный код Fedore Core можно обеспечить выпуск и поддержку линейки дистрибутивов Альт CoreOS и, возможно, Альт IoT.

В настоящий момент в Sisyphus портированы большинство пакетов Fedora CoreOS:

Для создания и продвижения дистрибутива Альт CoreOS необходимо:

- портирование недостающих пакетов или создание собственных аналогов;
- создание или использование существующего ПО для развёртывания дистрибутива в облачных средах, включая Yandex Cloud;
- организация процедура создания образов дистрибутива и «выкатки» его на сервера.

Таким образом создание и поддержка в рамках репозитория Sisyphus линейки дистрибутивов Альт CoreOS, Альт IoT вполне решаемая задача.

Литература

[1] Сайт Fedora CoreOS Documentation: `https://docs.fedoraproject.org/en-US/fedora-coreos/`

[2] Сайт Ostee Documentation: `https://ostreedev.github.io/ostree/`

[3] Презентация Fedora CoreOS Introduction — Dusty Mabe (Red Hat): `https://www.youtube.com/watch?v=JepNm7R0LSo`

[4] Российский разработчик операционных систем «Альт»: `https://www.basealt.ru/`

[5] Репозиторий Sisyphus: http://www.sisyphus.ru/

Пакет	**Описание**	**Sisyphus**	**Примечание**
ostree	Система обновления версий операционных систем на базе Linux	портирована	
ignition	Утилита для управления дисками во время начальной установки системы	портирована	
dracut	Инструментарий создания файловой системы начальной загрузки initramfs	портирован	В рамках ALTLinux используется собственный технологический процесс создания initrams
butane	Утилита конвертации файлов описания конфигурации из формата YAML в формат JSON Ignition	-	Достаточно простая утилита
coreos-installer	Программа установки CoreOs	-	Возможно создание собственной программы
rpm-ostree	Аналог программы apt-get установки RPM-пакетов в файловую систему ostree	-	Возможно потребуется не весь функционал

Николай Костригин
Обнинск, «Базальт СПО»

Доверенная загрузка GNU/Linux в режиме UEFI Secure Boot в 2021 году

Аннотация

В докладе рассматривается технология доверенной загрузки на базе UEFI Secure Boot и реализация поддержки этой технологии в дистрибутивах GNU/Linux.

Даётся историческая ретроспектива организации подписи stage1 загрузчика shim и обзор текущего состояния дел в этой сфере.

Приводятся требования к инфраструктуре подписи исполняемых файлов на примере репозиториев компании Базальт СПО.

Обсуждаются тенденции ужесточения режима Secure Boot и их возможное влияние на мир СПО.

Введение

EFI, а позднее UEFI является спецификацией описывающей расширяемый интерфейс загрузочной прошивки вычислительной машины, пришедшей на смену BIOS (Рис. 1).

Рис. 1: Позиция EFI в стеке компонентов вычислительной системы.

Причиной для выполнения подобной работы явились ограничения BIOS унаследованные ей от первых IBM PC: выполнение в 16-битном реальном режиме процессора, вытекающее из этого ограничение на объём используемой оперативной памяти и другие.

На данный момент UEFI поддерживается несколькими аппаратными платформами, в частности ix86 (ia32, x64), ARM/ARM64 (aarch32, aarch64), RISCV32/RISCV64, существует реализация для MIPS.

UEFI Secure Boot

Протокол Secure Boot (защищённая загрузка, далее «SB») стал опциональной частью спецификации UEFI. Он основан на ассиметричной криптографии и позволяет построить цепочку доверия от аппаратной платформы до запуска ядра операционной системы. Причиной для его появления называлась защита от выполнения недоверенного кода на всех этапах загрузки системы и попытка защититься от внедрения UEFI Boot kit, вредоносного ПО способного записать себя в энергонезависимую память и скомпрометировать загружаемую ОС.

Shim: stage1 загрузчик с поддержкой Secure Boot

Для решения проблемы запуска ОС на базе GNU/Linux было разработано минимум два загрузчика, которые в Microsoft согласились подписать — это PreLoader.efi от Linux Foundation, не получивший, однако, широкого распространения, и shim от RedHat и SUSE.

Shim распространяется под лицензией BSD-3-Clause и является компромиссом между необходимостью иметь подписанный загрузчик для режима SB у сообщества свободного ПО и нежеланием Microsoft подписывать, например grub2, распространяемый под лицензией GPLv3. [1]

Во многих реализациях UEFI-прошивок пока сохраняется возможность очистить заводскую базу доверенных сертификатов, заменив своими, или дополнить её.

Для полноценной поддержки доверенной загрузки в дистрибутиве GNU/Linux подписаны должны быть, как минимум, все стадии загрузчиков и ядро операционной системы. Для возможности запуска shim в режиме SB он должен быть подписан ключом Microsoft. Остальные стадии загрузки могут быть заверены самоподписным сертификатом поставщика ПО.

Shim-review и подпись файлов shim в Microsoft

Процедура подписи shim претерпела ряд изменений за прошедшие годы.

В 2012–13 годах процедура заключалась, по сути, в отсылке подписанного ключом разработчика *.cab-архива содержащего неподписанный shim.efi и получении zip-архива с подписанным файлом. Этот процесс, хорошо задокументирован в статье Михаила Шигорина [2].

В конце 2017 появилась дополнительная процедура проверки исходного кода загрузчиков, получившая название shim-review. В состав комитета, осуществляющего ревью кода вошли представители Debian, RedHat, SUSE, Ubuntu. Результаты проверки хранятся в соответствующем репозитории на GitHub [3].

Для доказательства соответствия представленных на подпись бинарных файлов shim исходному коду выполняется их тестовая пересборка, сравнение sha256 хэшей.

Для удостоверения организации и подписи отправляемых ей в Microsoft файлов она должна заказать EV (Extended Validation) Code Sign сертификат у одного из крупных центров авторизации (CA), что возможно только при наличии сведений о компании в одном из международных бизнес-справочников.

Уязвимости цепи доверенной загрузки и обновлённые требования процедуры shim-review

В 2020 году был обнародован набор уязвимостей в загрузчике grub2, позволявших обойти активированный Secure Boot и выполнить произвольный код загрузки на уязвимой машине. Он получил название «Boot Hole». Исправление всего списка CVE включало набор из 28 патчей. Этот инцидент привлёк внимание большого количества исследователей безопасности. В результате, спустя несколько месяцев, был обнародован ещё один набор уязвимостей, который стали называть Secure Boot Bypass 2021. Исправление для него включало уже 123 патча.

Чтобы исключить возможность загрузки ранее подписанных уязвимых версий grub2 необходимо внести изменения во всю цепочку доверия дистрибутива.

Масштаб проблемы связанный с двумя упомянутыми инцидентами вскрыл проблему в дизайне UEFI: для отзыва всех подписанных

копий загрузчиков потребовалось бы занять порядка половины объёма NVRAM отведённого под чёрный список DBX. В экстренном порядке был разработан улучшенный механизм отзыва недоверенных EFI-программ, который получил название SBAT — Secure Boot Advanced Targeting.

```
SBAT for shim:

sbat,1,SBAT Version,sbat,1,https://github.com/rhboot/shim/blob/main/SBAT.md
shim,1,UEFI shim,shim,1,https://github.com/rhboot/shim
shim.altlinux,1,ALT Linux,shim,15.4-alt2,http://git.altlinux.org/gears/s/shim.git

SBAT for grub:

sbat,1,SBAT Version,sbat,1,https://github.com/rhboot/shim/blob/main/SBAT.md
grub,1,Free Software Foundation,grub,2.06-rc1,https://www.gnu.org/software/grub/
grub.altlinux,1,ALT Linux,grub,2.06-alt1.rc1,http://git.altlinux.org/gears/g/grub.git

SBAT for fwupd has the same style

sbat,1,UEFI shim,sbat,1,https://github.com/rhboot/shim/blob/main/SBAT.md
fwupd,1,Firmware update daemon,fwupd,1.5.9,https://github.com/fwupd/fwupd
fwupd.altlinux,1,ALT Linux,fwupd,1.5.9-alt1,http://git.altlinux.org/gears/f/fwupd.git
```

Рис. 2: Пример исходных данных для генерации .sbat секции для доверенных EFI программ.

По сути SBAT представляет собой дополнительную ELF-секцию (.sbat) добавляемую к каждому доверенному EFI-загрузчику или EFI-программе и содержащую информацию о происхождении (версии апстримного кода), вендоре и поколении (версии релиза от момента внедрения SBAT вендорами) этого загрузчика (Рис. 2). В случае обнаружения уязвимости в одном из загрузчиков отзыв может быть осуществлён более адресно, без нерационального расходования ресурса NVRAM. Подробнее о SBAT можно прочитать в [4].

Итак, ключевыми отличиями обновлённой процедуры shim-review в 2021 году являются :

1. требование наличия .sbat секции в бинарных файлах;
2. требование замены встроенного сертификата или добавление всех ранее подписанных уязвимых бинарных файлов в чёрный список;

3. запрет подписи новым сертификатом старых (не содержащих исправления для Boot Hole + SB Bypass 2021) сборок grub;
4. ограничение списка по-умолчанию доверенных загрузчиков (фактически до связки shim->grub2->vmlinuz)
5. требование использования HSM для хранения ключей и сертификатов подписи.

Почему все дороги ведут в Microsoft?

Сертификаты, попадающие в UEFI firmware большинства IBM-PC совместимых машин, доступных на современном рынке, включают таковые от производителя конкретного аппаратного решения и от корпорации Microsoft, как поставщика наиболее часто предустанавливаемой ОС. Поставщиков компьютеров много, а поставщик ОС Windows — один. Это привело к ситуации, в которой общим сертификатом для большинства машин с поддержкой UEFI SB является именно сертификат Microsoft и корпорация оказалась в положении центрового игрока на поле доверенной загрузки PC-совместимых компьютеров. От решения её экспертов зависит сможет ли тот или иной программный продукт загружаться без отключения режима Secure Boot.

Справедливости ради, хотелось бы отметить, что общение как с представителями комитета shim-review, так и с представителями лаборатории Hardware Development Center Microsoft проходит исключительно в конструктивной и доброжелательной манере.

Требования к инфраструктуре сборочных серверов и репозитория

Для подписи бинарных файлов на сборочном сервере репозитория Sisyphus и его стабильных ветвей используется программа pesign. Инструмент создан RedHat и довольно широко распространён среди разработчиков дистрибутивов, развивается ими.

От инструмента получила своё название и группа разработчиков на сборочном сервере ALT Linux Team имеющих право подписывать загрузчики и ядро. Подзадания с такими бинарниками должны получить подтверждение (approve) от члена группы @pesign (Рис. 3). Невыполнение этого правила приведёт к тому, что загрузка маши-

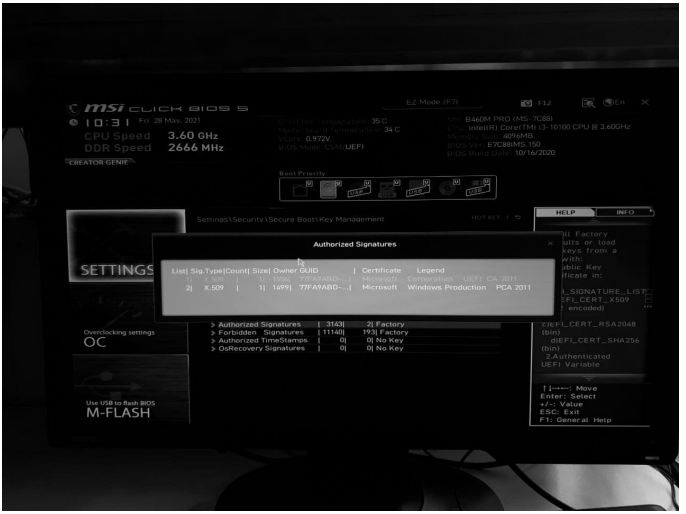


Рис. 3: Список доверенных сертификатов в разделе Authorized Keys UEFI одной из серийных PC-совместимых машин предустановленных при производстве.

ны с активированной опцией Secure Boot после обновления до версии пакета с неподписанным файлом будет нарушена.

Проблемы в реализациях UEFI прошивок

Нерешённой проблемой является то, что несмотря на открытость кода UEFI (Tianocore / edk2), исходные коды финальных проприетарных прошивок компьютеров реализующие эту спецификацию остаются закрытыми. При этом поведение кода прошивок тестируется, как правило, только на совместимость с MS Windows, а совместимость с другими ОС не просто не тестируется, но иногда даже открыто игнорируется.

Примером такой реакции может служить недавний отказ техподдержки Acer устранить проблемы в коде автоматического создания и выбора загрузочной записи в прошивке ноутбука Acer Swift 3. В процессе исследования проблем с загрузкой ноутбука была выявлена,

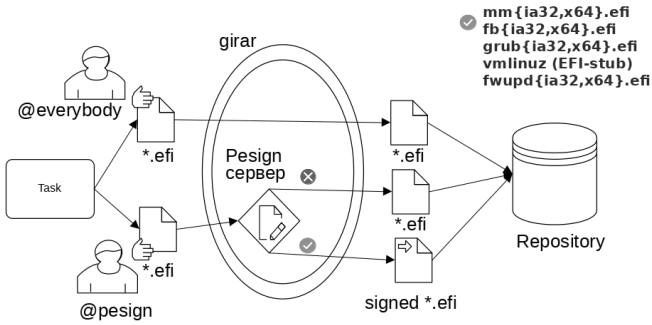


Рис. 4: Схема организации подписи бинарных файлов в инфраструктуре Базальт СПО.

на наш взгляд, серьёзная проблема: при составлении и отображении списка загрузки вместо текста из полей создаваемых в NVRAM в элементы списка попадает текст из случайных областей памяти. Рис. 5 иллюстрирует, как вместо опции загрузки отображается набор символов повторяющий имя производителя прошивки из заголовка окна. Исправления этой ошибки нет в течение более, чем 3 месяцев.

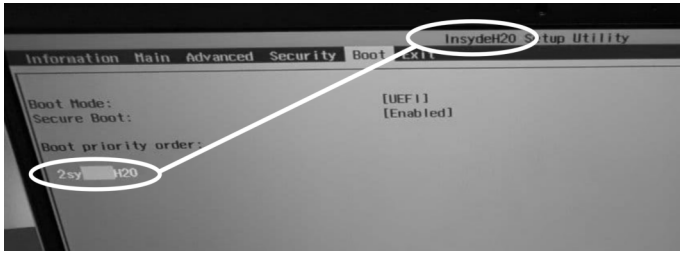


Рис. 5: Иллюстрация чтения произвольных областей памяти в UEFI прошивке ноутбука Acer Swift 3.

Также начинает прослеживаться тенденция невозможности отключения Secure Boot или замены доверенных сертификатов на поль-

зовательские в ряде моделей потребительских устройств выпускаемых в последнее время.

В некоторых прошивках заявленный функционал CSM реализован с приоритетом UEFI. Это приводит к тому, что в гибридных образах загрузчиков инсталлятора (на примере линейки ОС Альт) активация режима CSM (Legacy BIOS) всё равно приводит к старту машины в режиме UEFI и запуску соответствующего инсталлятора, если обнаружена загрузочная партиция EFI.

UEFI SB по ГОСТу

Существующая спецификация UEFI (самая свежая версия 2.9 от марта 2021 года) не содержит поддержки криптографии по ГОСТ (только DSA и RSA).

Однако, в число организаций, участвующих в UEFI Forum от России, входят ИСП РАН и Kraftway. Возможно они могли бы стать пионерами применения криптографии по ГОСТ, сначала в локальных версиях прошивок, а затем и предложив эти изменения в апстримный код.

Также для отечественных систем с локализованной версией UEFI разумно было бы встраивать ключи российских производителей и центра сертификации наряду с иностранными.

Заключение

Подводя итог всему сказанному, нужно отметить, что UEFI Secure Boot и её поддержка в ОС на базе GNU/Linux становится всё более зрелой, обнаруживаются и исправляются ошибки. Однако, наряду с этим, текущая реализация прошивок UEFI страдает от отсутствия диверсификации удостоверяющих центров, в том числе российских, а также закрытости исходных текстов проприетарных прошивок, что ограничивает доверие к ним.

Литература

[1] `https://techcommunity.microsoft.com/t5/hardware-dev-center/updated-uefi-signing-requirements/ba-p/1062916`

[2] `https://en.altlinux.org/UEFI_SecureBoot_mini-HOWTO`

[3] https://github.com/rhboot/shim-review

[4] https://github.com/rhboot/shim/blob/main/SBAT.md

[5] https://github.com/rhboot/shim-review/issues/165

Эльвира Хабирова, Александр Анисимов
Москва, ООО «Открытая Мобильная Платформа»
Проект: Trusted Firmware https://developer.trustedfirmware.org/w/

Проекты организации Trusted Firmware: свободное системное ПО обеспечения безопасности на процессорах ARM

Аннотация

На сегодняшний день практически все SoC для мобильных устройств основаны на процессорах архитектур ARMv7-A и ARMv8-A. Поскольку мобильные устройства зачастую хранят информацию, которая может представлять интерес для злоумышленников (такую как данные банковских приложений), мобильные SoC на базе процессоров ARM, как правило, поддерживают доверенную среду исполнения, предоставленную технологией ARM TrustZone.

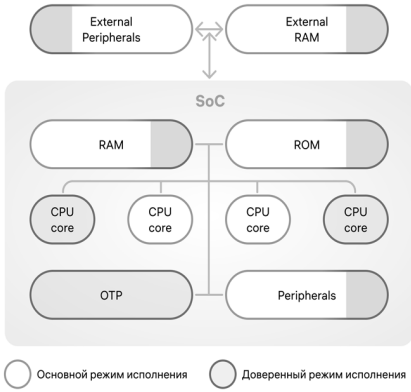
Данный доклад сфокусируется на проектах, находящихся под управлением организации Trusted Firmware; будет подробно рассказано, как эти проекты формируют целостный набор свободного системного ПО для доверенных режимов исполнения. Также будет рассказано о том, как российская компания «Открытая мобильная платформа» привносит свой вклад в это сообщество и его проекты.

Что такое TrustZone

Данные приложений, установленных на современных мобильных устройствах (МУ), представляют большой интерес для злоумышленников, так как позволяют получить доступ к важной персональной или бизнес-информации. Такими данными являются содержимое банковских приложений, биометрическая информация, ключи шифрования и электронной подписи. Задачей комплекса мер безопасности, применяемых в операционных системах МУ, является предотвращение угроз хищения или повреждения таких данных. Комплекс мер

противодействия угрозам тесно связан с использованием механизмов защиты, предоставляемых аппаратной платформой, и поддержкой данных механизмов со стороны операционной системы.

На сегодняшний день практически все SoC для мобильных устройств базируются на ARM-процессорах Cortex-A ARMv7-A/v8-A, и, как правило, обладают поддержкой технологии ARM TrustZone [1]. Эта технология позволяет организовать доверенную среду исполнения —Trusted Execution Environment (TEE), изолированную от основной операционной системы. Доверенная среда исполнения используется для запуска приложений, требующих повышенного уровня безопасности, например —биометрической идентификации, платёжных систем и т. д. [2]



ARM TrustZone основывается на аппаратном разделении ресурсов между двумя состояниями процессора — основным и доверенным. Помимо этого ARM предоставляет дополнительные IP-блоки контроллеров памяти и периферии — TrustZone Address Space Controller (TZASC), а также TrustZone Protection Controller (TZPC). Эти блоки позволяют ограничить доступ к определённым участкам адресного пространства или периферийным устройствам из обычного режима исполнения (основной ОС). Переключение между основным и доверенными режимами исполнения ядра процессора осуществляется путём вызова специальной инструкции — SMC (Secure Monitor Call).

С точки зрения высокоуровнего взаимодействия обмен данными между основной средой исполнения и доверенной, как правило, организуется на основе стандартизованного API по спецификации GlobalPlatform [3].

Семейство Cortex-A ARMv8-A имеет следующий набор режимов исполнения (Exception Levels): EL0 для исполнения пользовательских программ, EL1 — для исполнения ядра основной ОС, EL2 —уровень гипервизора, EL3 —монитор безопасности для переключения между нормальным и безопасным режимами исполнения; S-EL0 —для исполнения доверенных сервисов, S-EL1 — для исполнения доверенной ОС, S-EL2 —для доверенного гипервизора.

О проекте Trusted Firmware

Trusted Firmware — проект сообщества с открытым руководством [4]. Trusted Firmware предоставляет эталонную реализацию доверенной среды исполнения с открытым исходным кодом под лицензией 3-BSD для процессоров Armv8-A и Armv8-M. Кодовая база проекта поддерживается в соответствии с актуальными спецификациями Arm. Суммарно проекты под управлением TF реализуют полный набор программного кода для исполнения в режимах EL3 / S-EL2 / S-EL1 / S-EL0. Эти проекты в том числе служат целям дефрагментации кода, работающего на этих уровнях.

- Trusted Firmware-A и Trusted Firmware-M
  Эталонная реализация программного обеспечения уровня привилегий EL3 для Armv8-A, Armv7-A, Armv8-M. Берёт на себя процедуру загрузки доверенной ОС, а также предоставляет основной ОС и доверенной ОС некоторые сервисы во время работы системы —связанные с управлением питанием и поддержкой конкретного SoC и конкретной платформы.
  Официально поддерживается более 30 платформ от более чем 16 вендоров.

- OP-TEE
  Доверенная среда исполнения, исполнение на уровнях S-EL1 / S-EL0 —непосредственно реализует доверенную среду исполнения для использования совместно с ОС Linux в качестве основной ОС.

- Mbed TLS
  Библиотека, реализующая криптографические примитивы, работу с сертификатами в формате X.509 и протоколами SSL/TLS и DTLS. Подходит для использования во встраиваемых системах благодаря малому объёму кодовой базы. Её используют в том числе TF-A, TF-M, и OP-TEE.
- Hafnium
  Гипервизор S-EL2, на основе которого реализуется эталонный менеджер безопасных партиций (Secure Partition Manager, SPM) для систем, которые поддерживают расширение Armv8.4-A. Он позволяет иметь несколько изолированных безопасных партиций (SP), которые будут работать на уровне S-EL1.
- Trusted Services
  Фреймворк для разработки и развёртывания сервисов корня доверия устройства в различных средах безопасного выполнения, включая те, которые предоставляют OP-TEE и Hafnium, а также безопасные анклавы.

ОМП и Trusted Firmware

Российская компания «Открытая мобильная платформа» занимается разработкой ОС «Аврора» для мобильных устройств [5]. Операционная система «Аврора» включена в Единый реестр российских программ для электронных вычислительных машин и баз данных (Рег. No1543 от 05.09.2016) и соответствует требованиям по безопасности информации ФСТЭК России к операционным системам типа «А» четвёртого класса защиты, а также требованиям ФСБ России по защите информации классов АК2/КС2.

Как разработчик ОС для мобильных устройств, компания «ОМП» ведёт разработку и развитие собственного продукта для реализации функционала среды ARM TrustZone; данный продукт получил название «Аврора TEE». Основными направлениями функционала «Аврора TEE» являются: предоставление приложениям из основной ОС доверенных криптографических сервисов, управление ключами, безопасное хранение данных, а также аудит основной ОС и состояния устройства в целом как один из механизмов безопасности МУ.

В начале 2021 года «ОМП» вступила в сообщество Trusted Firmware. Открытость и прозрачность в управлении проектами TF

позволяют отслеживать изменения в кодовой базе и предлагать сообществу свои идеи, исправления и новые программные компоненты. Пребывание в сообществе помогает быть в курсе готовящихся изменений и вместе с сообществом определять пути дальнейшего развития. Также важно, что разработки «ОМП» (например, механизм плагинов для демона OP-TEE supplicant) и идеи (например, предложения по разделению аппаратных ресурсов между TEE и REE) не только развивают открытый проект и снижают затраты на поддержку, но и позволяют компании получать ценный опыт, что в итоге повышает качество всех продуктов «ОМП».

Литература

[1] TrustZone technology for Armv8-A. URL: `https://developer.arm.com/ip-products/security-ip/trustzone/trustzone-for-cortex-a`

[2] Антон Рыбаков. Контроль целостности мобильной ОС из доверенной среды исполнения // Системный Администратор — 2020 № 12 — стр. 26–30

[3] Global Platform Specifications Archive. URL: `https://globalplatform.org/specs-library/`

[4] Trusted Firmware. URL: `https://www.trustedfirmware.org/`

[5] Мобильная ОС Аврора. URL: `https://auroraos.ru/`

Илья Курдюков, Андрей Савченко
Новосибирск, Москва, Базальт СПО
`https://github.com/ilyakurdyukov/e2k-ports`

Оптимизация СПО для платформы Эльбрус

Аннотация

Рассмотрены различные методы оптимизации ПО для архитектуры Эльбрус. Представлены результаты оптимизации СПО проектов и даются практические рекомендации по подобной работе.

Введение

Эльбрус (e2000, e2k), в отличие от большинства других архитектур, использует набор инструкций с широким командным словом (VLIW) и позволяет выполнять десятки команд общего назначения за такт (до 25 для 4-го поколения, до 50 для 5-го). Это свойства приводят к гораздо более сильной зависимости производительности кода от степени его оптимизации как компилятором, так и программистом, с учётом особенностей архитектуры.

Рассмотрим способы оптимизации кода от простого к сложному.

Новые версии компилятора

Внутренний параллелизм и оптимальное заполнение VLIW — сложная задача, поэтому даже без изменения кода можно получить его существенное ускорение, используя последнее поколение компилятора. Например, прирост производительности между lcc-1.10 и lcc-1.25 составляет 2.05 раз для целочисленных вычислений и 3.58 раз для плавающей запятой[1].

Оптимизация для целевого процессора

lcc позволяет компилировать как приложения, работающие на любой архитектуре, начиная с заданной (`-march`), так и для конкретной модели процессора (`-mtune`). Во втором случае код получается быстрее, но он не будет работать на других моделях процессоров. Эксперименты с ядром показали прирост от 0.5% (мало, но статистически достоверно), до 8% в случае с 1C+.

Профилирование

Профилирование по семантике аналогично gcc (`-fprofile-generate`, `-fprofile-use`), но сложнее в практической эксплуатации. Для xz получен прирост $\sim$ 13%. Код доступен в Сизифе[2], где можно посмотреть пример практического применения и обхода проблем.

Использование SIMD

Для SIMD оптимизаций предпочтительно использовать интринсики (встроенные фукнции компилятора), а не писать всё на непосредственно на ассмеблере. Основные особенности:

Векторные команды x86

SIMD команды Эльбруса копируют векторные команды x86, но есть различия:

- Нет варианта некоторых команд, например `_mm_dp_ps()`, она эмулируется компилятором, поэтому будет работать медленно.
- Операция Shuffle на Эльбрусе может использовать таблицу в 2 регистра, поэтому можно написать более эффективный код.

Векторные интринсики x86

lcc понимает интринсики от MMX до AVX2, используя эквивалентные команды для Эльбруса. Если эквивалента не будет, то будет использована медленная эмуляция.

Особенности архитектур

v3,v4: 64-х разрядные регистры и поддержку почти всех SSE4.1 команд, одна векторная инструкция вычисляет только 64-бит результата. Медленно читает невыровненные данные, в т.ч. если данные выровнены, но компилятор об этом не знает.

v5: регистры 128-разрядные, появились эквиваленты векторных команд, которые обрабатывают 128-бит. Быстрее читает невыровненные данные.

v6: предполагается ускорение работы с невыровненными данными.

Выбор интринсиков

Удобнее использовать интринсики x86, чем Эльбруса:

- Их можно скомпилировать как для v3 так и для v5, для эмуляции SSE интринсика компилятор сгенерирует одну команду для v5 и две команды для v3. На v3 будет использовано в два раза больше регистров. Но регистров у Эльбруса много (256).

Зато не нужно писать отдельные версии кода для v3 и v5, что сокращает разработку.

- Оптимизацию можно разрабатывать и отлаживать на x86, лишь в финале проверяя на Эльбрусе. Небольшая часть вещей специфичных для Эльбруса выносится в макросы (которые эмулируют то же самое на x86).
- AVX команды эмулируются тем же образом, как SSE на v3. Использовать не рекомендуется из-за большого перерасхода регистров.

Зависимости между данными

Компилятор для Эльруса должен заранее знать об отсутствии зависимости между данными чтобы сгенерировать быстрый код. Например, для кода:

```
for (i = 0; i < n; i++) dst[i] = src[i] + x;
```

компилятор не знает, существует ли зависимость между данными (например, что `dst` будет указывать на `src + 1`). Поэтому вставит ожидание между записью dst и чтением src из следующей итерации. Чтобы этого не происходило, нужно указывать `#pragma ivdep` перед циклом.

Выравнивание данных

В lcc-1.25 не работают`__attribute__((aligned(N)))` или `__builtin_assume_aligned(ptr, N)`, но есть два способа этого добиться:

1. Опция lcc `-faligned`, но тогда любые указатели будет считаться выровненными, что неудобно и опасно в больших проектах.
2. Прямой сброс младших бит указателя: `ptr = (T*)((intptr_t)ptr & -8);` тогда LCC понимает что память выровнена. Но если придёт невыровненный указатель то не будет ошибки, но будут неправильные вычисления, или придётся добавить assert (что лишний код, замедлит при использовании в цикле). Также это лишняя операция AND, что критично в части кода (например циклы или макросы).

Оптимизированное СПО

На Эльбрус портированы или оптимизированы д14 популярных СПО проектов[4], включая ffmpeg, x264, libjpeg-turbo — всего 33 тыс. строк, 1.3 МБ исходного кода. Они уже доступны в Сизифе, ведётся взаимодействие с апстримами.

Аудио	mp3	aac	ogg	opus
base	1.17	1.26	1.57	2.62
mcst	1.30	1.40	1.76	2.84
open	0.98	0.94	1.23	2.23

Видео	mpeg2	mpeg4	xvid	avc	vp9	hevc
base	51.55	53.17	57.07	88.58	121.81	144.38
mcst	40.20	41.16	45.33	61.53	123.61	143.45
open	28.78	28.14	30.62	52.46	57.34	72.40

Результаты для основных кодеков в сёк (меньше — лучше) в режиме декодирования. Тестировалось на 720p, 201 сёк. base — исходный код, mcst — бинарные файлы МЦСТ, open — опубликованное открытое решение.

Литература

[1] Прирост производительности за счёт оптимизации компилятора
`https://www.altlinux.org/lcc#/media/File:Lcc-performance.jpg`

[2] xz: поддержка профилирования на e2k
`http://git.altlinux.org/gears/x/xz.git?p=xz.git;a=commit;h=7ed55a11d0643c5b53234c700163b72a41a7af30`

[3] Эльбрус upstream
`https://www.altlinux.org/РџРьСЁРçСЃСЋСЃ/upstream`

[4] Патчи оптимизации для e2k разного СПО
`https://github.com/ilyakurdyukov/e2k-ports`

Дмитрий Дегтярев
Саратов, ООО «Базальт СПО»

Проект: ADMC `https://github.com/altlinux/admc`

ADMC — графическое приложение для администрирования домена Active Directory

Аннотация

В докладе описывается приложение ADMC — инструмент для администрирования доменом Active Directory. Объясняется мотивация для разработки и отношение проекта к существующим решениям.

Задача

ADMC (Active Directory Management Console) это часть проекта интеграции групповых политик в ALT Linux. Для полноценного внедрения групповых политик требуется возможность администрирования доменом. Администрирование домена заключается в двух типах операций: просмотр и редактирование информации об объектах в домене. Просмотр информации включает в себя просмотр объектов, детей объектов, поиск объектов и просмотр атрибутов объекта. Редактирование информации включает в себя создание объектов, редактирование свойств объектов, перемещение объектов и т. д.

ADMC/ADUC

В Windows есть аналогичная программа ADUC (Active Directory Users and Computers) и ADMC разрабатывается в первую очередь как замещение этой программы. Это создаёт определённые ограничения для разработки, кратко: «ADMC должен быть таким же, как ADUC, но лучше». ADMC обязательно должен быть похожим по двум причинам. Во-первых, большинство будущих пользователей используют или пользовались ADUC. Это значит что у них сформировались определённые ожидания и привычки и им будет удобнее пользоваться ADMC если он будет похож на ADUC. Во-вторых, ADUC был выпущен в 2000 году, развивается давно, и до сих пор имеет большую базу пользователей. Следовательно, дизайн ADUC содержит много хороших идей проверенных временем. Поэтому в ADMC реализуется

большая часть дизайна ADUC с некоторыми дополнениями и улучшениями.

Реализация

Сложность этого проекта заключается в том, что ADUC является частью экосистемы Windows и его исходный код полностью закрыт. Также, ADUC использует Windows специфичные библиотеки и фреймворки, которые также полностью закрыты. В Linux есть альтернативы для некоторых частей: Qt, OpenLDAP, libkrb5 и другие. Для многих других функций альтернатив в Linux нет. Две самые большие части, которые нам пришлось реализовать самостоятельно, это Active Directory Services Interfaces (ADSI) и Microsoft Management Console (MMC).

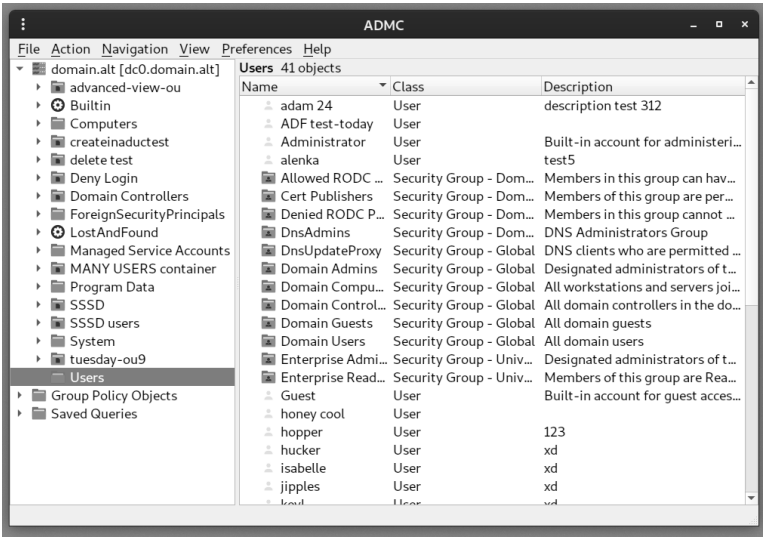


Рис. 1: Внешний вид главного окна приложения.

ADSI содержит в себе много функций для работы с доменом которые более удобны чем простые LDAP запросы. Грубо говоря, с помощью ADSI вы можете просто вызвать C++ функцию, которая изменит пароль пользователя. В OpenLDAP такой функции нет и вам

нужно будет написать её самостоятельно. Как это сделать правильно тоже придётся узнавать самим. Замещение ADSI в ADMC называется AdInterface.

MMC — это платформа для создания приложений администрирования в Windows. Многое из того что есть в MMC, доступно в Qt, но самой главной части которая называется «Консоль» в Qt нет. Консоль это супер-виджет включающий в себя почти всё что нужно типичному приложению для администрирования. В ней есть готовое дерево объектов и лист детей, меню бар и «Action» меню которое показывает операции для выбранного объекта. Консоль получилось сделать из частей из Qt, но с большим трудом потому-что функционал виджета консоли очень отличается от обычных Qt виджетов. Замещение консоли в ADMC называется ConsoleWidget.

Статус

На данный момент, в ADMC реализована большая часть функционала. Дальнейшая разработка включает в себя тестирование, визуальный дизайн. Также требуется доработка отдельных функций, таких как закладка «Безопасность» в диалоге свойств, диалог поиска и другие. Текущая версия приложения 0.6.2 и разработка ведётся активно. Пакет приложения входит в пакетную базу ALT Linux.

Литература

[1] Групповые политики / ADMC: `https://www.altlinux.org/Групповые_политики/ADMC`

[2] Active Directory Service Interfaces : `https://docs.microsoft.com/en-us/windows/win32/adsi/active-directory-service-interfaces-adsi`

[3] Microsoft Management Console 3.0 : `https://docs.microsoft.com/en-us/previous-versions/windows/desktop/mmc/mmc-3.0/ms692740(v=vs.85)`

Евгений Синельников, Игорь Чудов
Саратов, ООО «Базальт СПО»

Проект: Samba

Аналитика инфраструктурных решений службы единого каталога на базе Samba

Аннотация

Саратовской команде ООО «Базальт СПО» часто поступают вопросы по поводу инструментов управления доменом Active Directory. Совместными усилиями мы проделали огромную аналитическую работу по решениям для домена, а также прошли путь по выводу наших продуктов на международную арену.

Предисловие

Предпринимаются различные попытки решить проблемы построения и управления доменом. Так возникла FreeIPA, а также множество графических инструментов для отладки и управления базой данных LDAP. Тем не менее, не смотря на все усилия, множество этих инструментов не позволяют решить задачи, которые стоят перед пользователями Microsoft Active Directory.

Проблематика

Зачастую к решению задач управления доменом приходят в связи с выполнением требований импортозамещения. Данная тема требует повышения квалификации специалистов, которые занимаются построением корпоративных инфраструктур на базе Active Directory. В частности, требуется понимание того, что задачи должны решаться как со стороны сервера, так и со стороны клиента домена, и это задачи, которые имеют слабое пересечение.

Например, от сервера требуются возможности сквозной аутентификации машин и пользователей, работа с DNS, DHCP и почтой. Серверы, работающие как контроллеры домена, являются элементами, определяющими всю архитектуру сети предприятия, потому к их функционалу и надёжности предъявляются повышенные требования.

От клиентов требуется возможность обработки настроек, получаемый от системы управления конфигурациями. Когда речь заходит о домене, зачастую упоминают такой термин как «групповые политики». Правда, без понимания того, что оно из себя представляет.

Вывод

На рынке ПО для построения домена представлены очень ограниченные по функциональности решения в виде Samba или FreeIPA, и необходимо предельно чётко понимать, какого рода задачи необходимо решать на том или ином предприятии. Ошибка при проектировании инфраструктуры на начальном этапе и при составлении требований может вылиться в большие затраты на переделки в дальнейшем. ОС ALT поддерживает как домен Linux на базе Samba или FreeIPA, так и интеграцию с Active Directory, но в силу специфики ОС интеграция возможна при учёте ряда оговорок.

Литература

[1] Проект поддержки групповых политик в ОС ALT: `https://www.altlinux.org/Групповые_политики`

[2] Работа с контроллером домена в ОС ALT: `https://www.altlinux.org/ActiveDirectory/DC`

Игорь Власенко
, ALT Linux Team

Алгоритмы параллельной обработки пакетов для сборочницы дистрибутива

Аннотация

В докладе будут описаны алгоритмы параллелизации обработки заданий в сборочнице и параллелизации сборки внутри одного задания, позволяющие в десятки, а в некоторых случаях при наличии вычислительных мощностей и в сотни раз ускорить обаботку заданий в официальной сборочнице ALT Linux, а также будут обсуждаться психологические причины, мешающие их внедрению.

В настоящее время сборочница достаточно медленна и неудобна на задачах большого объёма, таких, как обновление perl или python, или при обработке большого числа мелких задач, к примеру — обновление java, которое выполняются последовательно и растягивается на многие часы, а то и сутки, хотя при хорошей параллелизации могли бы быть обработаны за несколько минут.

В то же время для большинства пользователей работа со сборочницей сейчас всё ещё находится в зоне комфорта. Можно предположить, что субъективная комфортная скорость работы сборочницы связана с его скоростью подготовки пакетов. Т.е. если человек за 8 часов подготовил 2 пакета и сборочница обработала их за 4 часа, то она достаточно быстрая. Если же человек за 8 часов подготовил 10 пакетов, и сборочница обработала их за 10 часов, то это уже субъективно медленная сборочница. Тот факт, что «крупнозадачные» пользователи страдают, показывает, что бутылочное горлышко пропускной способности сборочницы достигается слишком легко. У здорового дистрибутива пакетная масса растёт, и это только вопрос времени, когда в постоянных пробках окажутся и задания обычных пользователей. Альтернативно, плохая сборочница может стать (и становится, на примере p9-p10) фактором, тормозящим рост пакетной массы дистрибутива и, тем самым, угнетающим его развитие.

Общедистрибутивная сборочница по своей природе сложна для независимой разработки. Обычный опенсорсный проект можно форкнуть и пользоваться, сборочницу форкнуть можно, но пользоваться придётся старой, если не удастся убедить принять изменения. Чтобы не писать код «в стол», первым шагом надеюсь убедить в необходимости изменений. Хороший правитель не рубит голову гонцу, принёсшему дурные вести. Хороший полководец не пропускает сообщение, что передовые отряды терпят потери.

Сейчас можно проигнорировать запросы «крупнотоннажных» пользователей, рассудив что текущая сборочница по скорости устраивает большинство, и является критическим сервисом, работающим 24/7, поэтому к ней применим принцип «работает - не трогай». Это со временем приведёт к проблемам всего дистрибутива. А пока докладчик сейчас находится в положении «паникёров 1940-1941 года». Таких никто никогда не любил. Их было положено игнорировать и изолировать, чтобы не портили общую благостную картину «дружбы народов с отдельными недостатками», пока не грянула беда.

Михаил Быков
Москва, Диглосса
http://diglossa.org/

diglossa.js — инструмент медленного чтения

Аннотация

В докладе описывается приложение для чтения и редактирования многоязычных текстов на основе нового формата е-книг, .dgl, подобного формату .epub, но построенному на основе markdown, а не html

diglossa.js — десктопное приложение для Window, MacOS, Linux. Мощная читалка и редактор структуры любых е-книг (.epub, .pdf, .fb2, .csv, .html, .md, .txt), в том числе е-книг на основе собственного формата .dgl. А также генератор многоязычных контекстных словарей. Позволяет читателю самому создавать dgl-книги, добавлять новые переводы и генерировать новые словари.

diglossa.js — позволяет подключать обычные словари, и генерирует свои многоязычные контекстные словари на основе dgl-книг. На сайте http://diglossa.org вы видите примеры подобных словарей, сайт может использоваться как справочник и контекстный онлайн-словарь. Здесь же вы можете загрузить примеры книг в формате .dgl (пока только в виде файла).

diglossa.js — бесплатна, имеет открытый код, свободную лицензию GNU GPL, модульную структуру и интуитивно ясный интерфейс

diglossa.js — построена на основе формата электронных книг .dgl — подобного .epub, но имеющего в основе не html, но markdown

Плагины позволяют использовать любой язык, помимо широко распространённых

Основные возможности:

- импорт книг: .epub, .pdf, .fb2, .html, .md, .txt и — .dgl
- импорт словарей — .sd, .dsl
- автоматическое определение языка текста
- вызов соответствующего языку словаря по alt-mouse-move
- локальный и полнотекстовый поиск
- закладки
- импорт и параллельное подключение переводов той же книги

- редактирование структуры книги
- автоматическая проверка синхронизации абзацев
- экспорт синхронизированных книг в формате .dgl (пока только в виде файла)
- генерация многоязычных контекстных фразовых словарей

markdown

dgl — формат электронных книг, подобный формату epub (см. http://idpf.org), но имеющий два отличия. Во-первых, он использует не html, но (псевдо)-markdown. Во-вторых, он не имеет специального файла оглавления .toc, но использует саму разметку markdown для построения оглавления. Во всём остальном .dgl копирует .epub, его можно назвать версией .epub-markdown, все возможности .epub сохранены.

Псевдо-markdown — потому что из блоковых тегов используется только один — p, или paragraph. Последовательность (псевдо) абзацев включает заголовки, абзацы, строки таблицы, строки списка, примечания и всё подобное горизонтальное. Таким образом, текст книги превращается в последовательность абзацев. Благодаря этой особенности формата .dgl можно легко сопоставить два или более параллельных текста. В синхронизируемых текстах каждый абзац соответствует своей параллельной паре.

Синхронизация и авто-синхронизатор

Абзац может быть удалён, скопирован, разбит на части, etc — см. соответствующий раздел документации. Однако, при синхронизации в текст книги исправления не вносятся, но создаётся синхронизатор. Это позволяет в любой момент откатить внесённые изменения, даже вернувшись к редактированию книги после редактирования или чтения иной книги, или рестарта диглоссы. Авто-синхронизатор значительно ускоряет работу по проверке синхронизации абзацев подключаемой книги. При экспорте пакета книг их синхронизаторы теряются, и публикуется лишь окончательная форма книг.

Синхронизация абзацев книги — ручная работа. Но после синхронизации разделов, которая в любом случае должна выполняться вручную, возможно запустить автоматическую проверку синхронизации текстов. Реализованы два механизма авто-преверки синхронизации абзацев — сложный, с проверкой соответствия стемов абзаца, требующий коннекта с сетью и создания словарей соответствия стемов словоформ языковых пар. И простой, на основе сравнения длины

строки абзаца, числа фраз и пр. формальных признаков. Оказалось, что простой метод даёт неплохие результаты и вполне годен к применению в реальной практике. Собственно, авто-синхронизация есть убойная фича, делающая всё вышеописанное реальным.

Плагины

Диглосса группирует словоформы языка на основе стемов словоформ. Здесь стем — понятие не грамматики языка, а механизма обработки строк. Это просто подстрока, группирующая словоформы максимально плотно. А какие словоформы попадут в группу — не имеет значения. (Это не лингвистика, а обработка строк). В плагине реализуются два метода — анализатор и синтезатор. Анализатор в простейшей форме есть просто обычный стеммер. Всего могут быть три формы анализатора — a) стеммер, b) сложный стеммер, дающий два или несколько вариантов стема и c) полноценное разложение словоформы на все возможные цепочки последовательностей разбиения словоформы на подстроки. (На длинных словоформах это может быть весьма резурсозатратно). Затем происходит проверка наличия стемов или любых подстрок в словарях. Если метод a) не даёт результата, стартует метод b), затем метод c). В конце синтезатор собирает полученные результаты проверки в компактную форму. Диглосса вызывает анализатор и отображает результат, полученый от синтезатора, не имея представления о том, что происходит в плагине.

Ближайшие цели

- опубликовать пример плагина, и руководство по написанию плагинов для любого языка
- свести текст, параллельные тексты и справочный аппарат е-книги к единому, всегда актуальному источнику в системе контроля версий, автоматически клонируемому на компьютер читателя. (т.е. публикация в системе контроля версий, вдобавок к экспорту пакета в файл).
- дать возможность читателю самому добавлять новые параллельные переводы и справочные материалы к единому актуальному источнику. И, напротив, загружать себе лишь необходимые ему части е-книги. В том числе иметь возможность совмещать переводы и иные части книги, имеющие различные и несовместимые в обычной е-книге лицензии.

code & download

- описание проекта: `http://diglossa.org/`описание

- code: `https://github.com/mbykov/diglossa.js`
- download: `https://github.com/mbykov/diglossa.js/releases/latest`

На момент публикации есть только пакеты для Linux, в том числе для Alt Linux также.

Станислав Левин
Обнинск, Базальт СПО

Проект: Fleet Commander https://fleet-commander.org/

Fleet Commander и remote-viewer

Аннотация

Добавление поддержки альтернативного SPICE-клиента (remote-viewer) для Fleet Commander. Сложности реализации и дополнительные предусловия.

Введение в FC

FC позволяет легко администрировать профили окружений рабочего стола доменных пользователей. Для этого авторизованному на соответствующие манипуляции с данными администратору профилей достаточно один раз создать шаблон(VM), запустить этот шаблон в режиме живой сессии плагина Cockpit(web консоль), произвести необходимые изменения в графической сессии и сохранить изменения в профиль. Профили и политики хранятся в LDAP, а среди поддерживаемых контроллеров домена: Windows AD(только прямая интеграция) и FreeIPA. Применять профили можно для пользователей, групп пользователей, узлов и групп узлов.

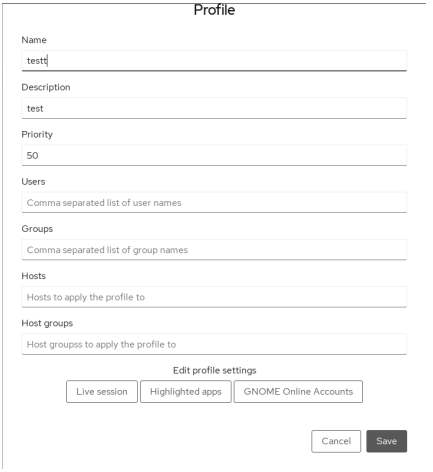
FC работает со следующими приложениями/подсистемами:

- `Gsettings`
- `LibreOffice`
- `Chromium`
- `Chrome`
- `Firefox`
- `NetworkManager`

FC состоит из трёх частей:

1. Административная часть — это FC admin, представляющий из себя плагин для Cockpit и служащий для:

- управления всеми профилями, хранящимся во FreeIPA или AD
- настройки подключения к libvirt узлу
- настройки профиля
- настройки типа SPICE клиента

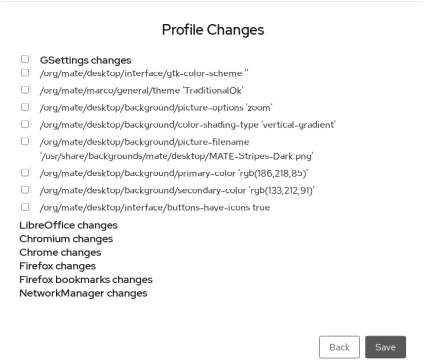


2. Регистратор изменений — это FC logger. Будучи установленным на шаблоне(VM) для профилей, в режиме живой сессии детектирует вносимые изменения в конфигурацию окружения пользователя и производит оповещение о наступившем событии FC admin. FC использует libvirt(KVM) для управления живой сессией в режиме snapshot (все изменения вносимые в VM существуют только пока она запущена).

В случае с FreeIPA профили хранятся в base64 закодированном JSON и с ними можно работать через CLI.

3. Клиентская часть — это FC client, применяет профиль пользователя.

При входе пользователя в систему, SSSD запрашивает профиль у соответствующего домен контроллера и оповещает о готовности FC client.



А что не так?

Режим живой сессии работает через компонент spice-html5. В официальной документации к этому продукту утверждается, что он работает только в современных Web браузерах, является медленным, отсутствует множество из функциональностей SPICE и, по сути, является прототипом. А рекомендовано к использованию virt-viewer. На практике это подтверждается, особенно на стендах с вложенной виртуализацией, где spice-html5 оказался непригоден к использованию. В то же время remote-viewer прекрасно справляется с задачей, поэтому было решено добавить поддержку remote-viewer как SPICE клиента.

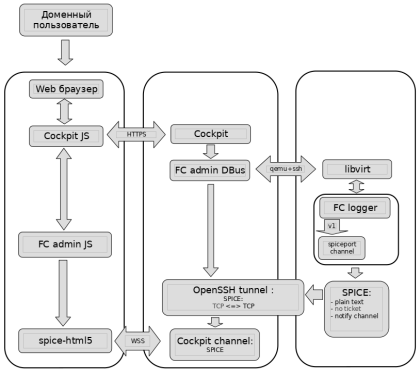
Текущая архитектура FC

Недостатки:

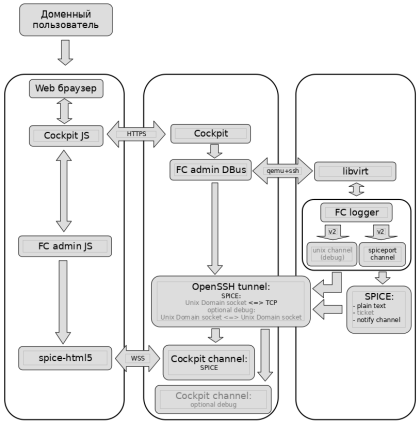
- открытые порты на libvirt и cockpit узлах
- незащищённая SPICE сессия
- нет возможности просматривать логи FC logger из Web браузера
- ограничение на максимальный размер изменения (FC logger отправляет JSON данные как есть)

Преимущества:

- единая точка взаимодействия(Web браузер)
- достаточно открыть один порт(HTTPS) на Cockpit узле для живой сессии FC



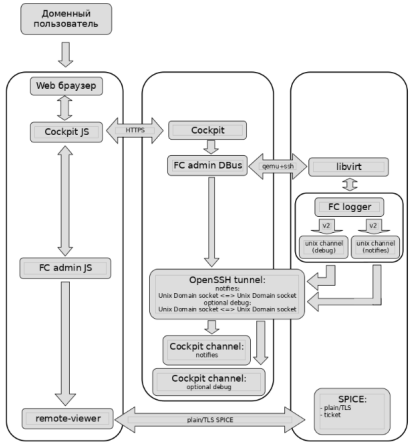
Улучшенная архитектура FC для spice-html5



Преимущества:

- нет незащищённых открытых портов FC на libvirt и cockpit узлах
- защищённая SPICE сессия(тикет)
- возможно просматривать логи FC logger в Web браузере
- базовый протокол для FC logger

Архитектура FC для remote-viewer



Недостатки:

- требуется дополнительная программа SPICE-клиент (remote-viewer)
- недостаточно открыть один порт для живой сессии FC(HTTPS) на Cockpit узле, требуется пул SPICE портов для libvirt узла.

Предусловия для изменений.

Планируемые изменения были значительны и почти наверняка вызвали бы регрессии. Поэтому обязательным условием принятия новой функциональности является покрытие её тестами. Однако, оказалось, что у проекта есть unit тесты, но нет CI(по крайней мере, публичного), где можно запустить эти тесты. Было решено добавить и CI.

Исходники проекта хостятся на github, поэтому выбор был между Azure Pipelines и Github Actions. Оба CI имеют как преимущества, так и недостатки по сравнению друг с другом, но схожы в одном — предоставление бесплатных worker'ов для opensource проектов(Azure — 10, GH Actions — 20). Но последний в то время был ещё в бета версии, поэтому остановились на Azure.

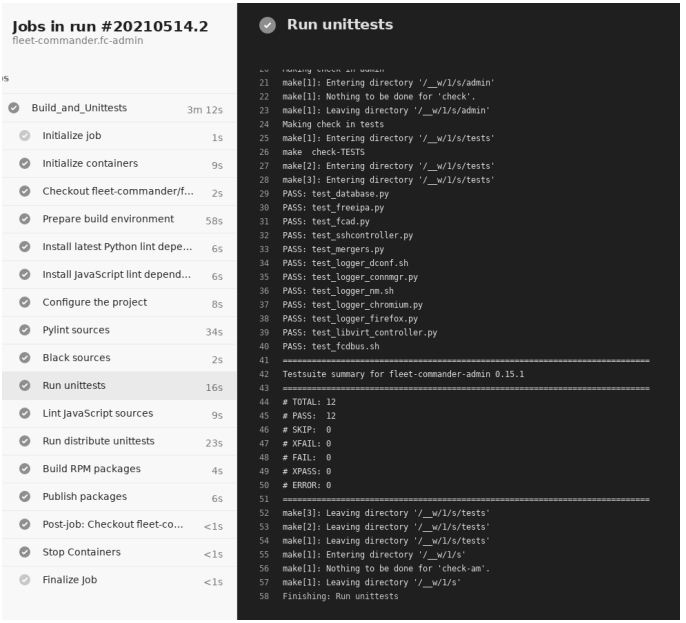
Код FC в основном Python и JavaScript.

Для Python были использованы:

- black — автоформатирование по PEP8
- pylint — статический анализатор

Для JS — eslint и плагины для него.

Данные инструменты помогли найти в коде множество проблемных мест и исправить их. CI запущен, можно двигаться дальше.



Принятие изменений и итоги

В сентябре 20-го года были приняты в апстрим патчи для CI, а в декабре изменения для remote-viewer. Важными итогами можно считать:

- поддержка remote-viewer как SPICE клиента
- поддержка как TLS, так и plain SPICE(не рекомендуется, было добавлено по просьбе апстрима)

- покрытие тестами новой функциональности
- исправление некоторых проблем с безопасностью
- адаптация и запуск публичного CI (Azure Pipelines)
- адаптация и применение инструментов для статического анализа кода(pylint, eslint)
- переход JS на систему модулей
- возможность просмотра вывода логгера с VM в консоли Web браузера
- исправление множества найденных ошибок

Андрей Михеев
Москва, ООО «Процессные технологии»
http://runawfe.ru/, http://www.runawfe.org/

Свободная система управления бизнес-процессами и административными регламентами RunaWFE Free. Изменения, вошедшие в последние версии

Аннотация
RunaWFE Free — свободная система управления бизнес-процессами и административными регламентами. В докладе описаны изменения, вошедшие в версии 4.4.1 и 4.4.2.

Система RunaWFE Free

RunaWFE Free — свободная система управления бизнес-процессами. Реализует процессный подход к управлению, в соответствии с которым деятельность предприятия представляется в виде набора выполняющихся экземпляров бизнес-процессов. Бизнес процесс содержит набор узлов, соединённых между собой переходами. По этим переходам перемещаются точки управления. В некоторых узлах они порождают задания, которые должны выполнить сотрудники или информационные системы.

Система позволяет повысить эффективность административного управления путём формализации повторяющихся последовательностей действий, а также за счёт возможности быстрого изменения бизнес-процессов в ответ на изменение условий бизнеса.

Изменения в версиях 4.4.1 и 4.4.2

Общая функциональность

В релизы вошли следующие изменения:

Начиная с версии 4.4.1 релизы содержат чат участников экземпляра бизнес-процесса.

Чат позволяет участникам экземпляра бизнес-процесса обмениваться друг с другом сообщениями во время исполнения экземпляра. Используется в случаях, когда для выполнения задания исполнителю не хватает данных или нужна какая-то дополнительная информация. При этом для получения этой информации не нужно изменять состояние бизнес-процесса или генерировать новые задания, так как получение этих дополнительных данных носит второстепенный характер и не изменяет бизнес-логику исполнения экземпляра бизнес-процесса.

Войти в чат можно из пункта меню «Чаты», форму задания или через свойства экземпляра бизнес-процесса (см. Рис. 1).

Рис. 1: Вход в чат через свойства экземпляра бизнес-процесса

Чат представляет собой иерархический список сообщений. Сообщения относятся сразу ко всей иерархии подпроцессов, порождённых корневым бизнес-процессом. Участниками чата могут быть пользователи, ранее выполнившие какое-либо задание экземпляра бизнес-процесса, а также все пользователи, входящие в любую из инициализированных ролей.

На основе существовавших ранее отельных элементов реализовано внутреннее хранилище бизнес-объектов.

Для работы с внутренним хранилищем был добавлен специальный элемент палитры — «Хранилище данных», связанный с задачей сценарий с помощью пунктирного перехода (см. Рис. 2).

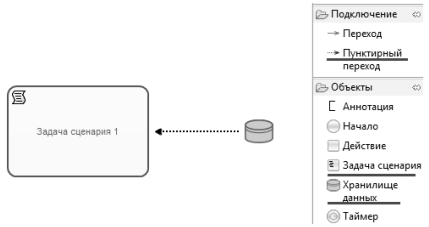


Рис. 2: Элемент палитры — «Хранилище данных».

По прямой связи «иконка — обработчик» (<-) в конфигурации доступна только операция SELECT. По обратной связи (->) доступны 3 операции: INSERT, DELETE, UPDATE (см. Рис. 3).

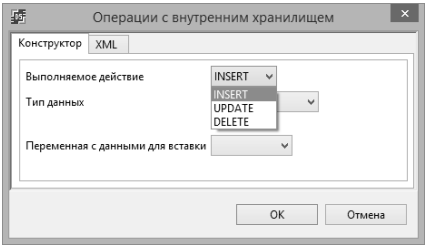


Рис. 3: Три возможные операции: INSERT, DELETE, UPDATE.

Другие изменения общей функциональности:

- В WS API, в сервис управления процессами и контроля их исполнения добавлены команды по работе с сигналами
- Для переменных добавлена валидация значений по умолчанию
- Добавлена возможность валидации параметров подпроцессов и мультиподпроцессов при запуске
- В дистрибутив для Windows добавлен сбор статистики

- Улучшен компонент выбора из списка пользовательских переменных

RunaWFE Free Server
Реализована отправка сигнала из веб интерфейса (см. Рис. 4)

Рис. 4: Отправка сигнала из интерфейса.

В настройки добавлена возможность отключения использования в системе прав на категории объектов.

Для упрощения работы и знакомства с системой, начиная с версии RunaWFE Free 4.4.1 по умолчанию отключена проверка прав доступа. Это означает что при доступе к объектам системы (исполнители, определения БП, экземпляры БП, отчёты, отношения, бот-станции, источники данных, действия с системой) права пользователя не проверяются. Настройки «Настройки» -> «Настройки прав доступа» позволяют включить или отключить проверку для определённой категории (см. Рис. 5).

Другие изменения RunaWFE Free Server:

- Добавлены права на отчёт
- Добавлена возможность сравнение версий бизнес-процессов в веб-интерфейсе аналогично «git diff»
- Реализована проверка бизнес-процесса на совместимость при обновлении определения бизнес-процесса в той же версии
- Добавлена возможность использовать список пользовательского типа с атрибутами, соответствующими колонкам SQL-таблицы, в качестве результата обработчика SQL
- Раздел с ошибками выполнения перенесён в отдельный пункт меню

Графический дизайнер

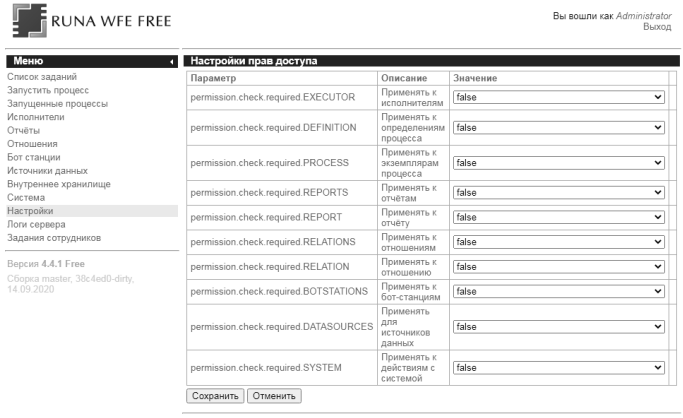


Рис. 5: Настройки, позволяющие включить или отключить проверку для определённой категории прав.

Реализовано следующее:

- Расширена контекстная палитра элементов графического дизайнера.
- Упрощена настройка конфигурации задачи-сценариев и ботов «Word: Формирование документа DOCX»
- Упрощена настройка конфигурации SQL-обработчика
- Реализована возможность непрерывающего возникновения событий / срабатывания таймера
- Добавлен фильтр по названию бизнес-процесса при импорте с сервера
- Реализована возможность смены типа события путём клика на иконку события
- Добавлен новый режим рисования элементов на схеме бизнес-процесса: если новая линия перехода «тянется» и завершается не на элементе, то открывается локальная палитра элементов для добавления нового элемента
- Добавлена возможность редактирования имени перехода через контекстное меню

- Добавлена команда «Переименовать» в контекстное меню узла-действия

Клиент-оповещатель о поступивших заданиях

- Добавлен фильтр по названию бизнес-процесса при импорте с сервера

Литература

[1] Ссылки на сайты проекта RunaWFE Free: `http://runawfe.org/rus` , `http://runawfe.ru`

Роман Бородин
Москва, «Лаборатория МБК»

Почтовый (многофункциональный) сервер Tegu

Преамбула

Вначале несколько строк о нашей компании. Во-первых, потому, что это наше первое выступление на конференции разработчиков СПО. А во-вторых, это позволит понять как мы пришли к разработке этого продукта.

Лаборатория МБК занимается системной интеграцией на базе свободных и отечественных решений Linux. Мы построили большое количество информационных систем полностью корпоративного уровня (тысячи пользователей) на базе Линукс. Сегодня в разных круг можно активно обсуждать и даже проводить робкие пилот-проекты на тему возможности миграции, а мы как раз те самые люди, которые на практике доказывают, что это а) возможно и б) в этом большой практический смысл. Впрочем, большинство здесь собравшихся разумеется знают это самостоятельно.

Однако, в процессе работы мы разумеется сталкиваемся с проблемами, для которых приходится искать решение. Одна из наших «болей» как интеграторов как раз и заключается в отсутствии почтового решения корпоративного уровня, способного конкурировать с лучшими импортными образцами.

Источники вдохновения

Сложность установки

Философия UNIX учит нас тому, что каждая программа должна выполнять одну единственную функцию, делать это хорошо и интегрироваться с другими программами посредством стандартных протоколов. Электронная почта — одна из первых глобальных служб, которые появились в интернете, она начала свою история в 1971 году. На тот момент ещё не было очевидно, что именно IMAP и SMTP победят и фактически останутся единственными для доставки почты. Как следствие сейчас мы имеет отдельные компоненты, из которых собираются современные свободные почтовые сервера.

Но на практике всё давно не так. Почта, календари, адресные книги, интерфейс-веб — всё это входит в состав одного сервиса, которые требуют пользователи. В своей программе мы постарались побороться с избыточностью числа компонентов, создав монолитный модуль.

Отсутствие должной поддержки служб каталогов

С интеграцией тоже всем привычные, но необоснованные проблемы. С одной стороны мы имеем данность, что серверы не в состоянии работать автономно (standalone) по собственной базе пользователей, с другой — интеграция с каталогами — это танец с бубном. Да, работает, но почему так сложно? И почему доступна интеграция только с одним каталогом? А если их несколько (что чаще всего и бывает на практике). В случае с Tegu вы можете использовать и собственную базу пользователей, для описания которой мы использовали открытый формат JSON, так и с каталогами разного типа. А точнее один сервер может работать с любым количеством источников авторизации в любой комбинации.

Такая архитектура позволила решить ещё один наболевший вопрос — делегирование административных полномочий. В нашем случае мы имеем не одного администратора, наделённого root-ом, а любое количество администраторов каталогов т.к. все настройки производятся именно через каталог.

Недостатки систем хранения

Ещё один исторически сложившийся, прекрасно себя зарекомендовавший, но мешающий жить, фактор — система хранения. Существует два наиболее популярных формата: mbox и maildir. Оба они хороши, тем, что не требуют дополнительного ПО для организации хранения, но на этом достоинства и заканчиваются.

Mbox с его блокировками файлов в принципе не пригоден для организации мультисерверной работы, maildir, осуществляющий доступ на уровне письма, в принципе пригоден, а в частности слишком неэффективен для поиска и массовых операций.

Наш сервер располагает двумя типами систем хранения. При этом один экземпляр Tegu в состоянии работать с несколькими хранилищами разного типа сразу.

Maildir без изменения стандарта модернизирован. Добавленный механизм кеширования позволил добиться кратного прироста производительности.

Второй способ — хранение почтовых сообщений и конфигурации сервера в базе данных PostgreSQL. Этот способ (и практика это доказала) позволяет получить ощутимый прирост производительности на высоконагруженных системах, а главное, выстроить грамотную масштабируемую системы на мультисерверных конфигурациях. Таким образом, мы получили возможность сколько угодно увеличивать мощность нашего сервера, при это обеспечивать его отказоустойчивость.

Отсутствие кластерных решений

В самом начале разработки мы ставили своей задачей создать не сервер рабочих групп типа Exim или Postfix, а сервер масштаба Яндекс. Сервер, в саму основу которого будет заложена возможность быстро и легко масштабироваться. Это нашло своё отражение и в архитектуре. В конфигурации кластера сервер хранит не только почту, но и всю свою конфигурацию в БД PostgreSQL. Это даёт возможность каждой из существующих вычислительных нод работать независимо друг от друга, принимая свою часть нагрузки. Что характерно, что и средства администрирования сервера могут быть задействованы через любую ноду. Внесённые изменения не нужно вручную или автоматически синхронизировать между нодами, они получают конфигу-

рацию из единой базы данных (которая в свою очередь тоже может быть построена по кластерному принципу).

Лицензирование

С точки зрения компонентов исходного кода все редакции сервера Tegu написаны так, что исключены отдельные ветки программирования (бранчи).

Следовательно, изменения, сделанные в любом модуле, автоматически применяются ко всем редакциям сервера. Корневой компонент, доступный под лицензией GPL, лежит в основе коммерческих сборок линейки Tegu Professional / Tegu Advanced / Tegu Enterprise. Таким образом очевидно, что мы относимся к разработке свободной версии точно так же ответственно, как и к разработке коммерческих модулей.

Дорожная карта

Важнейшая задача нашей компании – это непрерывное совершенствование наших продуктов. Это важно как для наших клиентов, применяющих наше ПО в своих системах, так и для сообщества, которое принимает участие в программировании свободно распространяемой редакции почтового сервера.

Вот главные направления развития 2021-го года:

1. Взаимодействие с антивирусным ПО по протоколу Milter (выполнено)

В текущей конфигурации сервер использует антивирусную проверку в режиме прокси-сервера, установленного перед ним. Данный способ не требует усилий по интеграции, но в ряде случае может оказаться неудобным. Например, присутствием различных механизмов для администрирования, отсутствием консолидированной статистики. Мы планируем сделать интеграцию по протоколу Milter, что в числе прочего позволит реализовать многопоточность и в случае необходимости распределить нагрузку. Таким образом, наш сервер станет ещё более отказоустойчивым.

2. Встроенная процедура собственного резервного копирования, восстановления

На самом деле собственная процедура позволит в диалоговом режиме настроить полное и/или инкрементное резервное копирование,

восстановление, а также миграцию хранилища с одного сервера на другой, либо с одного типа хранилища на другой. Это обусловлено тем, что вне зависимости от типа хранилища, формат бэкапа является универсальным.

3. Почтовое оповещение пользователей о критических событиях

Мы хотим сделать наш сервер более дружественным и предсказуемым для наших пользователей. Почтовое оповещение – это несколько сценариев, которые оповещают пользователя о состоянии его почты и почтового ящика. Например, приближение объёма к лимитирующей квоте, завершение выполнения рассылок, возможность восстановления из удалённого и т.п.

4. Публикация балансировщика Tiar

Балансировщик Tiar необходим в том случае, если в инфраструктуре пользователя нет балансирующего сетевого оборудования, но при этом Tegu эксплуатируется в мультисерверном (многонодовом) варианте.

Этот сервер будет опубликован под свободной лицензией GPL.

5. Поддержка механизма работы с сертификатами Let's Encrypt

В настоящий момент системный администратор сервера Tegu должен сам озаботиться приобретением сертификатов для сетевых подключений. Особенную неприятность доставляет ситуация, когда сертификаты по забывчивости оказываются просроченными. Мы хотим автоматизировать создание и применение сертификатов Let's Encrypt, которыми, при желании, может воспользоваться пользователь.

6. Правила обработки событий почтовых ящиков

В настоящий момент на сервере реализован механизм обработки входящей почты. Вы хорошо с ним знакомы. Однако мы хотим добавить ряд сценариев, которые выполняются не только в момент доставки, но в ряде других случаев. Например, заполненность ящика, удаление сообщения и пр. Нам кажется, что это может оказаться полезным.

7. Динамические папки на стороне сервера

Некоторые почтовые клиенты (Apple Mail, Evoluition) умеют создавать динамические папки, которые по сути являются запомненными фильтрами, которые отображаются в виде папок. Этот инструмент помогает быстро разобраться во всём обилии почты, а потому очень популярен.

Однако, далеко не все почтовые программы умеют создавать такие динамические папки. Наша задача заключается в том, что бы создать такую возможность на стороне сервера. В таком случае использовать эту функцию можно будет на любом клиенте.

Заключение

Мы будем благодарны вам за предложение по улучшению сервера и конечно приглашаем сообщество к совместной работе над сервером.

Дмитрий Винокуров
Пермь, Miro

Проект: FOSS News
`https://github.com/Gim6626/foss-news-gathering-server`,
`https://github.com/Gim6626/foss-news-tools`

Дайджесты FOSS News и средства автоматизации их подготовки

Аннотация

В мире свободного и открытого ПО и железа происходит много важных событий: релизы; внедрения; выпуск обзорных, аналитических и обучающих материалов; интервью с интересными людьми; объявления о мероприятиях и курсах; привлечения финансирования и прочее. В январе 2020 г. Пермской группой пользователей GNU/Linux (PermLUG) был начат проект сбора информации о перечисленных событиях в виде еженедельного дайджеста на «Хабре» под названием «FOSS News». С того момента выпущено более 70 дайджестов, настроен автоматический сбор материалов из нескольких десятков англо- и русскоязычных источников и разработаны прочие средства автоматизации, вокруг проекта сформировалась аудитория в несколько тысяч человек. Следующая статья посвящена достигнутым результатам, разработанному вспомогательному ПО и планам на будущее.

Концепция

Идея дайджестов возникла на встрече Пермской группы пользователей GNU/Linux (PermLUG) 23 января 2020 г. (сама группа существует с 1997-го года). Обсуждали идеи проектов и заметили, что

существует много FOSS (англ. Free and Open Source Software — свободное и открытое ПО) интернет-изданий, но зачастую материалы там имеют достаточно узкую направленность, а более общие издания содержат мало информации о FOSS проектах. Также изданий много и в них легко потеряться и пропустить что-то важное. Автором было предложено делать дайджесты каждую неделю, в которых была бы объединена как информация для широкого круга читателей, так и разделы со ссылками по более узким направлениям.

Категории дайджестов со временем сложились следующие:

- «*Главное*» — короткие описания со ссылками о нескольких наиболее важных для широкой аудитории материалах, это могут быть, например, новости о значимых внедрениях, особо интересные аналитические статьи или особо важные релизы;
- «*Короткой строкой*» — по трём категориям (новости, статьи, релизы) объединено всё, что было собрано подходящего из всех источников и не попало в главное, здесь просто заголовок и ссылка и всё разделено по подкатегориям (для разработчиков, системное, data science и прочие);
- «*Прочее*» — ссылки на другие дайджесты или что-то ещё, что не подошло под остальные категории.

В начале каждого дайджеста — аннотация и оглавление.

Дайджесты размещаются на «Хабре» как крупнейшем русскоязычном IT ресурсе.

Процесс подготовки

После нескольких первых выпусков был начат процесс автоматизации сбора информации и составления дайджестов, так как источников подключалось всё больше и вручную работать становилось слишком затратно по времени. После нескольких итераций автоматизации процесс подготовки выглядит так («[А]» — автоматические действия, «[Р]» — ручные, «[П/А]» — полуавтоматические):

А *сбор и фильтрация материалов* — робот на Python работает на удалённом сервере и несколько раз в день проверяет множество RSS и HTML источников, собирает заголовок, ссылку, дату и в зависимости от широкой или узкой специализации источника отфильтровывает по ключевым словам или берёт всё, сохраняет в базу;

П/А *дополнительная фильтрация и категоризация* — с помощью интерактивного консольного клиента-категоризатора, тоже на Python, вручную обрабатывается то, что собрал робот, отбрасывается то, что по ошибке прошло фильтрацию, распределяется по категориям и подкатегориям с помощью подсказок скрипта, которые тоже формируются по базе ключевых слов, всё это отправляется обратно на сервер в базу;

П/А *дополнение* — через административный интерфейс веб-фреймворка Django добавляются материалы из источников, не участвующих в автоматизированном сборе (в том числе из чата PermLUG);

П/А *формирование единого документа* — локальный скрипт выгружает с сервера из базы все заголовки материалов и ссылки для текущего выпуска и генерирует HTML;

Р *загрузка для совместного редактирования* — HTML загружается на Google Drive;

Р *доработка* — делаются переводы англоязычного текста, добавляются изображения для главных тем дайджеста и пишутся аннотации для всего выпуска и главных материалов;

Р *ревью* — ссылка на Google Drive отправляется для ревью в PermLUG и сторонним участникам подготовки дайджестов;

П/А *конвертация в окончательный формат* — после ревью документ выкачивается и ещё одним скриптом преобразуется в формат Хабра, в том числе создаётся оглавление;

Р *релиз* — код статьи загружается на Хабр, добавляются картинки и делается публикация;

Р *распространение* — аннотация и ссылка на дайджест добавляются в Telegram, RSS и по соцсетям (ВКонтакте, Fediverse, Facebook, Twitter);

П/А *аналитика* — собирается статистика (ещё одним скриптом).

Код инструментов автоматизации, как сервера «FOSS News Gathering Server» так и клиентских вспомогательных утилит «FOSS News Tools» — под свободными лицензиями: AGPL-3.0 и GPL-3.0 соответственно. Теоретически, после некоторых доработок этот код можно использовать для любых дайджестов.

Немного цифр, относящихся к подготовке:

- 30 англо- и русскоязычных источников;
- 1530 ключевых слов;
- ~2097 материалов поступает на отбор в неделю;
- ~303 материала проходят автофильтры;
- ~176 материалов (из которых десяток-другой об одних темах) проходят ручной отбор и попадают в дайджест.

В команде, помимо робота и автора статьи, ещё 3 человека, они помогают с вычитыванием.

Результаты

С 23-го января 2020 г. до момента написания этой статьи:

- выпущено 72 дайджеста, каждую неделю без перерывов;
- дайджесты просмотрены 205 615 раз на «Хабре» суммарно;
- в среднем на дайджест 2 855 просмотров на «Хабре»;
- на канал в «Telegram» подписался 481 человек.

Планы

Чтобы снизить нагрузку рутинной работой на автора статьи, открыть возможность уделять больше времени описанию контекста собираемых материалов и прочим улучшениям качества дайджестов, планируется реализовать распределённую подготовку дайджестов, разработав в дополнение к однопользовательскому консольному интерактивному категоризатору многопользовательский Telegram-бот.

Алексей Андреев
Москва, «Открытая Мобильная Платформа»
https://gitlab.com/AuroraOS/demos/asr-stt-project

Распознавание речи на мобильных устройствах, управляемых Linux, на примере ОС Аврора

ОС Аврора является POSIX-совместимым дистрибутивом Linux, спроектированным для решения корпоративных и бизнес-задач. Основная область применений устройств на ОС Аврора — это корпорации в России. Например, такие устройства используются полевыми сотрудниками для получения задач и отправки отчётов. Формирование подобных отчётов происходит с помощью заполнения сложных форм с большим количеством полей ввода и опций для выбора. Распознавание речи позволяет существенно ускорить этот процесс.

Есть и другие задачи, которые требуется решать разработчикам корпоративного прикладного ПО для мобильных устройств. На их основе можно сформулировать пользовательские требования к решениям: поддержка нескольких языков (по крайней мере, русский и английский языки), распознавание речи на устройстве (без необходимости использования сторонних серверов), скорость распознавания, не уступающая ручному вводу. Для реализации подобных функций в прикладном ПО удобно использовать готовые решения для распознавания речи.

Для POSIX-совместимых операционных систем доступны несколько СПО-библиотек, предоставляющих функции распознавания речи. Учитывая необходимость их использования, в том числе, в коммерческих проектах для ОС Аврора, сформулирован ряд дополнительных требований, связанных с инструментами сборки, особенностями хранения модели данных, активностью проекта и лицензией, допускающей достаточно свободное использование.

Удобным решением для задачи, удовлетворяющим критериям, является движок Kaldi. Однако, он зависит от математических библиотек, написанных на языке Fortran. А актуальные версии ОС Аврора на данный момент не поддерживает этот язык программирования в силу оптимизаций инструментов сборки. Требования к Fortran появляются, например, при использовании математических библиотек,

таких как LAPACK или OpenBLAS. Другие популярные проекты, которые от них зависят: PyThorch и TensorFlow.

Поэтому потребовалось разработать способ, который позволяет использовать зависимости Kaldi, реализованные на Fortran, в окружении, которое его не поддерживает. В докладе предлагается подход, позволяющий собрать программное обеспечение сторонним набором инструментов, и описываются ограничения по интеграции в окружение ОС Аврора.

Предложенный способ позволяет в сторонних решениях реализовывать распознавание речи на мобильных устройствах с помощью библиотеки Kaldi. Полученное решение удовлетворяет как пользовательским, так и техническим требованиям. Кроме того, предложенный подход может быть использован и в других случаях, когда Fortran требуется для сборки или запуска ПО, но окружение операционной системы или инструменты сборки не предоставляют возможности его использовать.

Валерий Синельников
Саратов, ООО «Базальт СПО»

Проект: alterator-dbus
http://git.altlinux.org/gears/a/alterator-dbus.git

Служба alterator-dbus, как возможность представить API модулей центра управления системой ALT через D-Bus

Аннотация

В докладе представлен инструмент, предоставляющий программный интерфейс для реализованных программ системного управления (модулей Alterator) на шине D-Bus. С помощью данного инструмента осуществляется регистрация сервиса на шине D-Bus, в котором модули Alterator представлены как отдельные объекты. Каждый объект предоставляет интерфейс для взаимодействия с модулем.

Для управления настройками ALT Linux есть возможность воспользоваться центром управления системой. Он позволяет в графическом интерфейсе управлять наиболее востребованными настройками системы. Центр управления системой состоит из нескольких неза-

висимых диалогов-модулей. Каждый модуль отвечает за настройку определённой функции или свойства системы.

Каждый модуль состоит из бэкенда и интерфейса. Бэкенд пишется обычно на «bash». Центр управления — это прежде всего способ объединить разрозненные модули. В отдельный процесс выделена часть, занимающаяся обслуживанием запросов к бэкендам (alteratord), организация доступа к которому может быть осуществлена с помощью разработанной службы `alterator-dbus`.

Доступ к функционалу модулей alterator через D-Bus

Разработанная служба регистрирует сервис, который предоставляет объекты модулей alterator на шине D-Bus. Для этого необходимо сформировать файл интроспекции для конкретного модуля, описывающий доступные методы. Файл помещается в /usr/share/dbus-1/interfaces/. После чего (после рестарта alterator-dbus.service) на D-Bus в системной шине в службе ru.basealt.alterator добавится новый объект с интерфейсом, описанным в файле интроспекции.

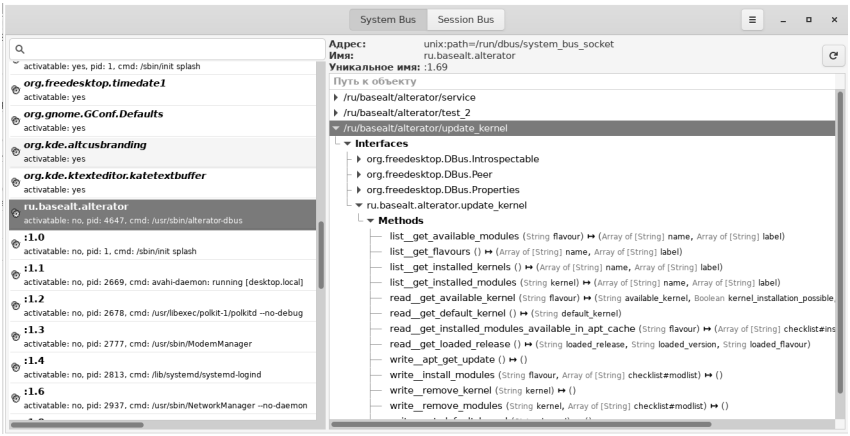
Предоставление непривилегированным процессам возможности доступа к alteratord

Для доступа к модулям через alteratord необходимы права администратора (root). В предоставляемом D-Bus интерфейсе контроль привилегий осуществляется с помощью правил (PolicyKit Authorization Framework). По умолчанию, для всех методов влияющих на состояние системы, осуществляется проверка на наличие административных привилегий у обратившегося через D-Bus пользователя. Для не влияющих на состояние системы методов доступ, по умолчанию, предоставляется без административных привилегий. При необходимости, правила по умолчанию для конкретных модулей можно прописать дополнительно. Также есть возможность конкретизировать правила доступа для отдельных методов.

Возможности использования alterator-dbus

Ключевые возможности, предоставляемые службой:

- Функциональность модулей центра управления системой ALT может быть представлена с помощью alterator-dbus через уни-



версальную систему IPC путём написания интроспекции D-Bus и, при необходимости, правил доступа.

- Использование D-Bus для доступа к функционалу модуля Alterator даёт возможность иметь представление о способах взаимодействия с ним средствами интроспекции самой шины, в которой отображаются имена методов и имена параметров.
- Использование в alterator-dbus polkit даёт возможность гибко регулировать доступ к предоставляемой функциональности.

Иван Савин
Саратов, ООО «Базальт СПО»

Проект: alterator-update-kernel
`http://git.altlinux.org/gears/a/alterator-update-kernel.git`

Обновление ядра в ОС «Альт»

Аннотация

На протяжении многих лет инструментом для обновления ядра в ОС «Альт» остаётся утилита **update-kernel**. Данное положение вещей выглядит естественным для опытных пользователей, но является

устрашающим для новичков. Реакция некоторых людей которые привыкли смотреть в окна, когда они видят чёрный экран и белые (зелёные) буквы на нём — широко открытые глаза и тихое дыхание. Это стресс. Последствия этого стресса могут помешать человеку нормально жить и стать пользователем Linux. Для того, чтобы помочь новичкам нужен новый, графический инструмент.

Модуль панели управления системой Alterator, называемый «Обновление ядра» реализует функционал утилиты update-kernel. Данный модуль предоставляет возможность просмотреть список установленных ядер, удалять модули для выбранного ядра, сделать ядро загружаемым по умолчанию или удалить ядро.

Функциональные возможности alterator-update-kernel

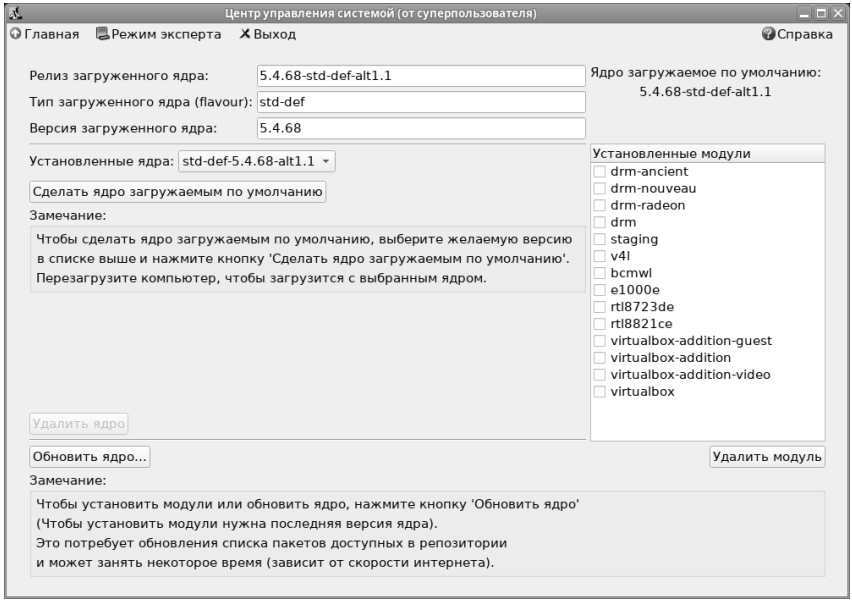
В дистрибутивах ALT существует возможность устанавливать несколько ядер одновременно. Ядра бывают нескольких типов по назначению (стандартное, нестабильное, специализированные). При обновлении операционной системы ядра автоматически не обновляются. Для обновления ядра применяется специальный консольный инструмент `update-kernel`, который позволяет обновить текущее ядро, или установить ядро заданного типа.

Сделать ядро загружаемым по умолчанию

После установки или обновления ядра старые ядра не удаляются. В случае возникновения проблем с новым ядром можно переключится на установленное ранее. Для этого можно выбрать нужное ядро в списке «*Установленные ядра*» и «*Сделать ядро загружаемым по умолчанию*».

Удаление ядра

Накопленный при обновлениях набор ранее установленных ядер можно удалить для освобождения дискового пространства. Для этого можно выбрать нужное ядро в списке «*Установленные ядра*» и «*Удалить выбранное ядро*».



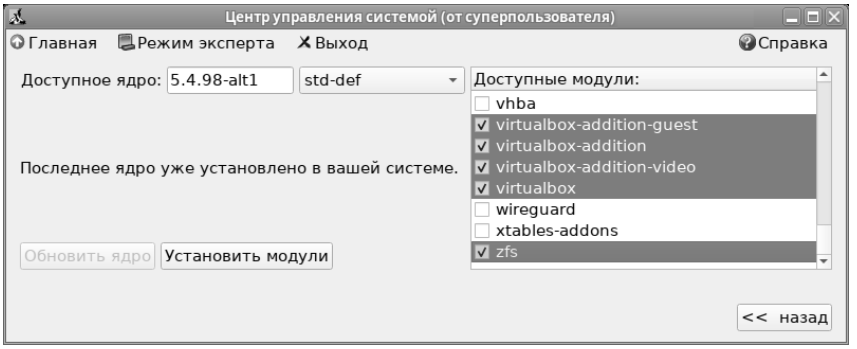
Удаление модулей

При установке операционной системы автоматически устанавливаются модули для различных аппаратных средств, включая различные модели видеокарт. Для уменьшения нагрузки при обновлениях неиспользуемые модули можно удалить. Для этого можно выбрать нужное ядро, модули которого нужно удалить, в списке «*Установленные ядра*». Затем, можно выделить модули, которые будут удалены, в меню «*Установленные модули*» и «*Удалить выбранные модули*».

Установка/обновление ядра и установка модулей

При обновлении ядра, обновляются все модули ядра собранные в виде отдельных пакетов, которые в себя включают дополнительные драйвера. Для того, чтобы обновить/установить ядро или установить модули ядра, необходимо перейти во вкладку «*Обновить ядро...*». При этом локальная база данных пакетов будет синхронизирована с

удалённым репозиторием, это может занять некоторое время. В открывшемся окне будет показано доступное к установке ядро. В выпадающем списке можно выбрать тип ядра (flavour). **std-def** — стандартное и основное ядро дистрибутивов ALT. В окне «*Доступные модули*» отмечаются модули, которые будут установлены. Чтобы «*Обновить ядро*» требуется подтвердить действие. Установленное ядро станет загружаемым по умолчанию.



Если ядро не требует обновления, в окне «*Доступные модули*» можно отметить модули ядра, необходимые к установке, и нажать кнопку «*Установить модули*».

Литература

[1] «Модуль Alterator-update-kernel» `https://www.altlinux.org/Alterator-update-kernel`

Леонид Кривошеин
Москва, ООО «Базальт СПО»
Проект: ALT Linux Team https://www.altlinux.org/

Инструменты начальной загрузки и массового развёртывания ОС Альт

Аннотация

Обзор стека инструментов для развёртывания ОС Альт: что и как могут использовать разработчики уже сейчас. Обзор новой системы загрузки altboot, призванной заменить propagator в дистрибутивах Альт. Комбинация новых методов загрузки с инструментами массового развёртывания.

ALT Rescue — основа системы массового развёртывания, спасательный LiveCD-дистрибутив умеренного размера, включающий богатый набор инструментов для работы с дисками и файловыми системами.

Rescue Launcher — заставшим MS-DOS напомнит назначение `C:\autoexec.bat`. В отличие от rc.local, процесс запускается интерактивно, а не в фоновом режиме, на конкретном терминале, сразу после запуска всех служб. По завершении этого процесса пользователь попадает в обычную командную строку. Разработан с таким расчётом, чтобы можно было автоматически смонтировать ОТКУДА-ТО некий каталог, содержащий ВАШ_СКРИПТ (или иной исполняемый файл), и запустить его. Параметр загрузки ядра `autorun` управляет тем, где и что ищется, каким образом и куда монтируется.

system-backup — автоматика для снятия образов уже установленной системы. Достаточно смонтировать носитель, на который эти образы будут сохранены. Работает на всех поддерживаемых аппаратных платформах.

usermode-fs-tools — набор утилит для работы с файловыми системами ext2/3/4 и vfat в режиме простого пользователя, как с обычными архивами, форматировать, подбирать нужный размер, дополнять их файлами, заполнять /dev, назначать любые права и владельцев, устанавливать системный загрузчик. Призван заменить tar2fs в mkimage-profile. В состав входит утилита *iso2stick*: она перепаковывает дистрибутивы ALT из ISO-9660 в образы USB-дисков. Поддерживается для всех аппаратных платформ.

По запросу крупных OEM-сборщиков мы начали ориентировать свои инструменты массового развёртывания на мастер-образы, что позволит применять специализированное оборудование для быстрого клонирования.

propagator — компактная программа, целью которой является определение местонахождения второй стадии инсталлятора, livecd или rescue-образа.

make-initrd — инструмент для сборки образов initramfs, используемых для начальной загрузки ОС Linux. *pipeline* — фича make-initrd, предлагающая новый «пошаговый» принцип загрузки. *bootchain* — форк и дальнейшее развитие pipeline, на момент написания предлагает с десяток модулей и ещё не попал в апстрим.

altboot — новая система загрузки, построенная поверх bootchain, фактически замена propagator в будущих дистрибутивах Альт, обеспечивающая новые функции и более гибкие возможности дальнейшего расширения.

Модули bootchain

core: базовый функционал, включает шаги debug, mountfs, overlayfs и rootfs
getimage: метод загрузки образов по сети (http/ftp) утилитой wget
waitdev: метод подключения локального носителя по заданной спецификации
interactive: интерактивное взаимодействие, предоставляет диалоговые виджеты
altboot: общий функционал и шаг, транслирующий аргументы пропагатора
localdev: методы загрузки disk/cdrom, установка дополнительных модулей ядра
liverw: дополнительный функционал для работы с сессиями LiveCD и Rescue
nfs: метод загрузки с сервера NFS
cifs: метод загрузки с сервера SAMBA

Основные шаги altboot

download: многофункциональный шаг, использующий curl, обеспечивает методы url, http, ftp, загружает в память squash, может загрузить образ чего угодно прямо в указанное устройство

checksum: подсчитывает контрольную сумму ранее загруженного или указанного

iso9660: монтирует устройство как CD/DVD-привод ISO-9660

squashfs: монтирует устройство или файл из каталога как squashfs

liveboot: многофункциональный шаг финальной стадии загрузки

Обычный /proc/cmdline:

```
root=bootchain bootchain=fg,altboot automatic=...
```

Необычный /proc/cmdline:

```
root=bootchain bootchain=waitdev,waitdev,fg,altboot \
                     waitdev=LABEL=alt-live-storage \
                     waitdev=CDROM: automatic=...
```

Переменные BC можно сложить в `/etc/sysconfg/bootchain`. Параметр `bc_debug` даёт расширенную отладку, отладочный журнал в stage1 доступен по Alt-F3, а также в `/var/log/bootchained.log`, при переходе в stage2 он копируется в `/dev/bootchain/bootchained.log`.

Чтобы написать свой модуль altboot, необходимо разобраться в порядке отработки «хуков». В этом помогут `git grep use_hooks` и скрипты в `altbot-mixed/hooks/*.sh`, получаемые при запуске скрипта `mix-altbot`.

Ссылки

1. `https://www.altlinux.org/Rescue`
2. `https://www.altlinux.org/Rescue/Launcher`
3. `https://www.altlinux.org/Rescue/Recovery`
4. `https://www.altlinux.org/Usermode-fs-tools`
5. `https://www.altlinux.org/Propagator`
6. `https://www.altlinux.org/Make-initrd`
7. `https://github.com/osboot/make-initrd`

8. https://www.altlinux.org/Make-initrd-propagator
9. https://lists.altlinux.org/pipermail/make-initrd/

Зоричев Борис, Попов Юрий
Москва, Национальный исследовательский университет «Высшая школа экономики»
https://inlnk.ru/WGMDK

Разработка протокола связи для устойчивой передачи видео в самоорганизующихся сетях

Аннотация

Данная работа посвящена разработке протокола связи для передачи видео в самоорганизующихся сетях. Представлен новый протокол связи, который позволяет передавать пакеты в видеопотоке в самоорганизующихся сетях, состоящей из 4 компьютеров. В процессе работы были проанализированы существующие проблемы самоорганизующихся сетей, а также варианты передачи информации в них.

Введение

В настоящее время все существующие решения для беспроводных самоорганизующихся сетей (WSN, mesh, MANET) способны передавать только текстовые сообщения и небольшие файлы, прикреплённые к этим сообщениям. Тем не менее для того, чтобы новые разработки в области беспроводных самоорганизующихся сетей стали успешными, необходимо, сделать возможным передачу мультимедийной информации. Среди всех типов мультимедиа необходимо сосредоточиться на передаче голосовой информации и видео потоков. Главной проблемой стандарта open80211s является многочисленные потери пакетов начиная со второго участка маршрута, которые делают невозможным передачу видео.

Новый протокол связи

Внести изменения в программный код mesh сети очень сложно, так как он работает по чётко определённому стандарту и связан с другими модулями. Поэтому для тестирования был разработан

новый протокол связи — протокол ближайшего соседа.

Для тестирования его работоспособности была собрана следующая экспериментальная установка

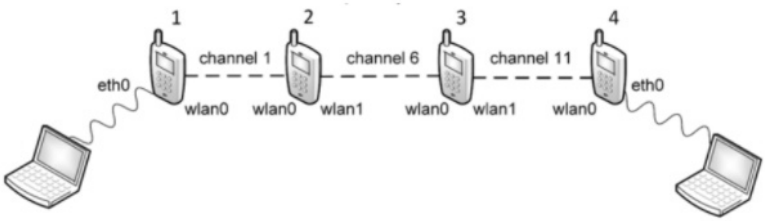


Рис. 1: Схема установки

Четыре узла, на которых инсталлировано программное обеспечение, реализующее беспроводную самоорганизующуюся сеть, расположены последовательно. Причём каждый узел может установить связь только с ближайшими соседями.

Наше программное обеспечение можно разделить на две логические части. Первая часть реализует функцию связи, то есть обработку и пересылку пакетов между соседними узлами. Вторая часть отвечает за маршрутизацию. Ниже приведены типы пакетов, используемые в нашем протоколе.

В качестве адресов узлов используются MAC адреса соответствующих устройств. При передаче нагрузки ad hoc кадрами текущий узел маршрута всегда знает MAC адрес следующего узла.

Нами было проведено тестирование нашего протокола для сети. При этом оборудование и схема связи остались такими же как для пакета open80211s.

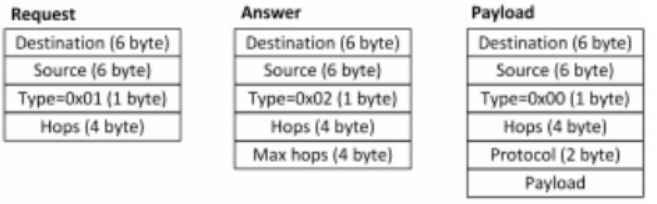


Рис. 2: Типы пакетов, используемые в протоколе

Название протокола	Количество хопов	Минимальная задержка сети, мс	Средняя задержка сети, мс	Джиттер, мс	Потеря пакетов, %
open80211s	1	1,7	5,5	3,6	0
	2	3,0	7,0	5,8	4
NH protocol	1	3.8	5.6	1.8	0
	2	7.6	9.6	3.6	1

Рис. 3: Результаты эксперимента

Литература

[1] Исходный код программы — `https://inlnk.ru/WGMDK`

[2] Руководство по тестированию — `https://inlnk.ru/qm2Rp`

[3] Mayhoub S. et al. Video Streaming in Ad Hoc Networks //2020 IEEE Applied Imagery Pattern Recognition Workshop (AIPR). – IEEE, 2020. – С. 1–5.

[4] Hiertz G. R. et al. IEEE 802.11 s: the WLAN mesh standard //IEEE Wireless Communications. – 2010. – Т.17. – №. 1. – С. 104–111

[5] Rath M., Pattanayak B., Pati B. A contemporary survey and analysis of delay and power based routing protocols in MANET //ARPN Journal of

Engineering and Applied Sciences. – 2016. – Т. 11. – №. 1. – С. 536–540.

[6] E. S. Sagatov, A. M. Sukhov, and P. Calyam, «Influence of distortions of key frames on video transfer in wireless networks», in 2010 5th International Symposium On I/V Communications and Mobile Network. IEEE, 2010, pp. 1–4.

[7] E. S. Sagatov and A. M. Sukhov, «Assessment technologies of communication quality in MANET networks», in 2019 10th IFIP International Conference on New Technologies, Mobility and Security (NTMS). IEEE, 2019, pp. 1–5.

Шатров Григорий
Москва, ФГАУ НИИ «Восход»
https://nsud.info.gov.ru/

Использование Open Source-подходов при построении Государственных ИС

НИИ Восход участвует в работах по созданию и сопровождению НСУД (Национальная система управления данными) с 2019. НСУД это важная, государственная ФГИС, которая будет неразрывно связана со СМЭВ посредством подсистемы обеспечения доступа к данным СМЭВ (ПОДД СМЭВ).

В 2021 году был заключён и выполнен контракт на Модернизацию ФГИС «ЕИП НСУД» в части выделения программного обеспечения компонента «Витрина данных» и его доработки, необходимой для подготовки к публикации на условиях открытой лицензии. В рамках этих работ были подготовлены: код Витрин данных (клиентской, «ведомственной» части НСУД) для публикации и документы «Соглашение о разработке и тестировании», «Регламент деятельности Комитета», «Государственная открытая лицензия».

Таким образом, НСУД является отличным примером того, как могли бы зарождаться и развиваться проекты с открытым кодом в органах государственной власти. Продуманная политика в области Open Source может дать:

- Повышение цифрового суверенитета страны;

- Появления в России конкурентноспособных коммерческих продуктов;
- Повышение информационной безопасности (ИБ) государственных ИС;
- Экономическую выгоду: Обмен кодом между множеством отечественных разработчиков;
- Контроль критически важной инфраструктуры.

Не трудно заметить, что всё, перечисленное ранее, укладывается в политику импортозамещения, проводимую в нашей стране с 2015 года. И действительно, пожалуй, главным преимуществом использования открытого кода может стать проведение широкого импортозамещения, без кратного увеличения затрат на информационные технологии в госорганах.

Одним из инструментов для использования потенциала Open-Source может стать «Государственная открытая лицензия», разработанная в рамках проекта НСУД. Лицензия разработана как с учётом международного опыта, так и на основе отечественных попыток создания лицензий, с привлечением соответствующих экспертов.

Хочется верить, что НСУД станет флагманским проектом в области «государственного открытого кода», а работы НИИ Восход получат высокую оценку в российской ИТ-отрасли. И, конечно, нам необходимо большее вовлечение в проект разных людей, а для этого, с учётом публикации кода на условиях открытой лицензии, есть все предпосылки.

Алексей Турбин
Рязань

RELZ4 — ускоренный алгоритм разжатия

Аннотация

Рассмотрены модификации алгоритма сжатия LZ4, которые обеспечивают ускоренное разжатие. Ускорение достигает $1.4\times$.

Введение

LZ4 — быстрый алгоритм сжатия, не использующий энтропийного кодирования. Сжатый поток имеет байтовую структуру, которую в упрощённом виде можно представить так:

```
[T|LLL|MO] [T|LLL|MO] ... [T|LLL]
```

Квадратные скобки показывают *последовательность* (sequence) из двух операций: копирование *литералов* и копирование *совпадения* (match). Первый байт последовательности `T` — это *токен*, в котором два 4-битных поля: количество литералов `T>>4` и длина совпадения `4+(T&15)` (мин. длина 4). Признак `(T>>4)==15` указывает на то, что литералов может быть больше 15, тогда считывается дополнительный *байт кодирования длины* (между `T` и `LLL`, не показан; аналогично может быть увеличена длина совпадения). Литералы `LLL` (переменное количество) копируются из сжатого потока в выходной поток. После этого копируется совпадение. Два байта `MO` (match offset) указывают *смещение назад* (относительно текущего указателя в выходном потоке), по которому находится совпадение. Самая последняя последовательность не полная, копируются только литералы.

Далее обсуждаются некоторые модификации, приводящие к увеличению скорости копирования.

Минимальное смещение

Когда копируется совпадение по маленькому смещению `MO`, участки памяти могут перекрываться. В таком случае нельзя использовать

обычный цикл копирования, похожий на `memcpy`, потому что копирование имеет побайтовую семантику (например, копирование по смещению 1 реплицирует последний байт). В LZ4 выполняется проверка `MO<16` (или `MO<8`): в таком случае совпадение нельзя копировать при помощи SIMD-регистра (соотв. 64-битного регистра), а требуется сложная процедура репликации начального участка. В [1] показано, что такая репликация может быть выполнена с помощью SIMD-инструкции перестановки байтов `pshufb`.

Однако можно просто не связываться со слишком маленькими смещениями, а использовать смещение `MO+C`, где `C=8` или `C=16`. Наши эксперименты показывают, что при `C=8` потери в сжатии составляют около 0.1%, а при `C=16` — ещё 0.3%. Такие потери почти всегда приемлемы. Но здесь нужно быть осторожным: низкие потери обусловлены использованием медленного алгоритма сжатия LZ4HC, который перебирает несколько совпадений (в поисках более длинного). А быстрый алгоритм сжатия просто отбросит единственное имеющееся совпадение из-за короткого смещения, так что потери в сжатии могут быть выше.

Настало время уточнить, что же бы оптимизируем. Мы оптимизируем скорость распаковки исходя из того, что данные сжимаются для долгосрочного хранения сравнительно медленным алгоритмом, а разжиматься будут многократно. Существует и другой сценарий: данные сжимаются, передаются и разжимаются только один раз. В таком случае бесполезно оптимизировать разжатие, а нужно оптимизировать сжатие, поскольку даже быстрое сжатие в разы медленнее разжатия.

Кодирование длин

Когда `(T>>4)==15` (или когда `4+(T&15)==4+15`), считывается дополнительный байт кодирования длины литералов (соотв. совпадения). Если этот байт равен 255, то считывается ещё один байт, и т. д. Такая система неэффективна: длина n кодируется $O(n)$ байтами. Вместо это можно было бы использовать кодировку VByte, в которой старший бит служит признаком продолжения. Длина кода будет $O(\log n)$, но длины 128-254 будут кодироваться двумя байтами.

В LZ4 байт длины имеет одно специальное значение, а в VByte половина всех значений специальные. Наилучший результат получается, когда специальных значений меньше половины. Для первого бай-

та мы выделяем 16 специальных значений, так что распаковка длины начинается так:

```
len = *src++;
if (len >= 256-16)
    len += 16 * *src++;
```

Для второго байта мы оставляем 32 специальных значения. Теперь длины 128-239 требуют одного байта, а два байта кодируют длины до 3823.

В LZ4 длины не ограничены, но алгоритмы сжатия ограничивает общий размер `LZ4_MAX_INPUT_SIZE` менее 2 Гб. В нашей модификации длины ограничены — чуть менее 4 Гб.

Длина совпадения 19

При кодировании совпадении с длиной `4+15` компрессор записывает дополнительный байт длины, равный 0. Это хуже, чем записать совпадение длиной 18 и оставить последний байт для следующей последовательности. В худшем случае последний байт станет литералом в следующей последовательности, но при распаковке не придётся считывать байт длины (который даст нарушение предсказания перехода). В лучшем же случае это байт станет частью нового совпадения.

В нашей модификации вообще исключена возможность кодирования длины совпадения 19: к длинам с дополнительным байтами прибавляется 20.

Конвейеризация

Посмотрим ещё раз на последовательность `[T|LLL|MO]`. Чтобы прочитать смещение `MO`, нужно сначала узнать количество литералов `LLL`. Из-за такого рода зависимости по данным процессор не может начать обработку совпадения параллельно с обработкой литералов. Можно было бы фиксировать расположение смещения: `[T|MO|LLL]`. Но в наших экспериментах это не приводит к увеличению скорости распаковки. Вместо это мы обнаружили, что основным препятствием является считывание токена в начале каждой итерации. Желательно, чтобы к началу следующей итерации токен был уже загружен и частично обработан. Это требует изменения формата сжатого потока, который теперь выглядит так:

```
[T] [T|LLL|MO] [T|LLL|MO] ... [T|LLL]
```

Здесь `[T]` является *сигнальным токеном*, а внутри последовательности кодируется *следующий токен*. Такая техника оптимизации называется *программная конвейеризация* (software pipelining). Последний токен реинтерпретируется как литерал.

Скорость разжатия

Для оценки скорости мы используем программу `lzbench` на микроархитектуре Haswell. Скорость распаковки LZ4 на корпусе Silesia — около 2.6 Гб/с, нашей модификации — 3.8 Гб/с (скорость `memcpy` — 8 Гб/с, память DDR3). В обоих случаях используются похожие алгоритмы медленного сжатия.

```
lzbench 1.8 (64-bit Linux)  Intel(R) Celeron(R) CPU G1840 @ 2.80GHz
Assembled by P.Skibinski

Compressor name   Compress. Decompress. Compr. size  Ratio Filename
memcpy             8077 MB/s  7968 MB/s   211938580 100.00 12 files
lz4hc 1.9.3 -7       31 MB/s  2677 MB/s    78102177  36.85 12 files
relz4 0.1            28 MB/s  3853 MB/s    78126965  36.86 12 files
```

Литература

[1] Алексей Миловидов. Как ускорить разжатие LZ4 в ClickHouse. Блог компании Яндекс, 21 мая 2019. `https://habr.com/ru/company/yandex/blog/452778/`