АНО «Институт логики, когнитологии и развития личности»
ALT Linux

Шестая конференция разработчиков свободных программ на Протве

Обнинск, 27–28 июля 2009 года

Тезисы докладов

Москва,
Институт Логики,
2009

В книге собраны тезисы докладов, одобренных Программным комитетом Шестой конференции разработчиков свободных программ. Круг рассматриваемых тем весьма широк: от новейших системных и прикладных разработок до правовых проблем, вопросов организации работы в проектах и аналитики.

Программа конференции

27 июля

10.30–12.00: Регистрация в холле ЦИПК

12.00–12.30: Кофе

Дневное заседание
12.30–16.45

12.30–13.00: Открытие конференции

14.00–14.45: Обед

16.45–17.15: Кофе

Вечернее заседание
17.15–19.00

28 июля

Утреннее заседание
10.30–11.30

11.30–12.00: Кофе

Дневное заседание
12.00–16.45

Вечернее заседание
17.15–19.00

Руслан Хихин Москва, НПО Агат, ALT Linux
Проект: Sisyphus

Перенос технологии «Сизиф» в смежные области

Уже более семи лет фирма ALT Linux использует репозиторий «Сизиф», занимаясь разработкой и выпуском дистрибутивов GNU/Linux. Под репозиторием обычно понимается хранилище пакетов. Репозитории имеют и некоторые другие компании, занимающиеся разработкой дистрибутивов Linux: OpenSuse, Fedora, Debian. Репозитории обычно используются в системах управления версиями, в них хранятся все документы вместе с историей их изменения и другой служебной информацией. В репозитории Сизиф хранятся исходные тексты всех программ и скриптов, а также изображения и другие файлы, применяемые в дистрибутивах, выпускаемых сообществом ALT Linux.

Сизиф — самый крупный проект по разработке свободных программ, созданный на основе русскоговорящей команды. Сизиф полностью открыт, т. е. не существует секретных патчей и закрытого тестирования с подписками о неразглашении результатов. Репозиторий Сизиф в каком-то смысле уникален — в нём самом содержится все технологии, которые в нём-же применяются.

С учётом приведённых особенностью репозитория Сизиф возникает идея применения технологий, используемых в его построении, для ведения локальных репозиториев предприятий — от репозитория отдельного заказа до ведения репозитория программного обеспечения и документации всего предприятия. Данный доклад посвящён проблемам, возникающим при попытке реализации этой идеи.

Постановка задачи

Представим проект программного продукта, имеющего в своём составе N-е количество задач и документацию к ним. Необходимо отметить следующие особенности разработки программного продукта:

- За каждую задачу отвечает конкретный программист, и только он должен иметь право менять его код.

- У каждого программиста есть начальник, который имеет право просматривать программный код и вносить свои предложения по его улучщениюа.
- Документация к программе может разрабатываться отдельным человеком, не всегда тем-же программистом, который разрабатывает программу.
- Программный код может компилироваться в различных операционных системах. т. е. сама сборка программного кода происходит вне репозитория. Но программый код должен хранится единообразно внутри репозитория (в виде src.rpm-пакетов). Для ОС на основе Linux из них впоследствии стандартным образом получаются бинарные пакеты. Для программного кода для Windows-проектов и для программной документации включается упрощённый вариант хранения кода.
- В последнее время сборочная среда Сизифа активно интегрируется с git-репозиториями. Хотелось бы при хранении пакетов документации и текста программ активно применять его преимущества с тем, чтобы пользователь мог сформировать запрос к репозиторию и получить нужную ветку или состояние на заданную дату.
- Для документации хорошо-бы иметь совместимость с какимнибудь Wiki.
- Для пользователя репозитория должен существовать простой и понятный ему способ положить программный код в репозиторий и получить код заданного релиза.

Проблемы реализации

При сравнении технологии Сизифа и проекта локального репозитория предприятия можно выявить, что на сегодняшний день не все потребности возможно реализовать в Сизифе. В частности, нет соединения технологии Сизифа и технологии Wiki. Также нет простого интерфейса между пользователем и репозиторием. Исходя из потребностей пользователя, необходимо создать простое кроссплотформенное приложение, которое позволит загружать заданный проект (в виде архива или src.rpm) и получать заданную версию исходного кода пакета для дальнейшей работы.

Можно отметить, что этим приложением должен быть графический кросплотформенный клиент ssh со специальными возмож-

ностями. Следует учесть, что для работы системы необходим acl-доступ, аналогичный имеющемуся в Сизифе. Для системы документации хорошо-бы иметь возможность предоставлять доступ к ней через Wiki.

Пути реализации

Несмотря на наличие реальных проблем, можно отметить, что создание локального репозитория предприятия на основе технологии Сизифа реально, но требует определённой доработки решений. Можно отметить, что эти решения возможно использовать в самом Сизифе для снижения порога вхождения в состав мэйнтейнеров репозитория и упрощения использования новых технологий, таких как git и gear.

Литература

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

Дмитрий Багаев, Артём Евсяков, ГОУ ВПО «Ковровская государственная технологическая академия имени В. А. Дегтярёва»

Проект: Pegas http://kpribor.edu.ru

Разработка свободного программного обеспечения для управления робототехническим комплексом

Аннотация

В статье рассматривается разработанный авторами программный комплекс управления учебным робототехническим комплексом (УРТК). Программный комплекс обладает следующими преимуществами: удобный пользовательский интерфейс с возможностью демонстрации управления роботом; возможность работы в режиме клиент-сервер; возможность удалённого управления через протокол TCP/IP, а также через выделенный интернет-канал с применением мобильного телефона; кроме того, он имеет уникальное средство по групповому

управлению роботами семейства УРТК. На комплекс получено свидетельство ОФАП.

Удалённое управление имеет широкий круг применения — от систем видеонаблюдения и систем «умный дом» до управления и мониторинга сложных технических объектов и комплексов. Целью данной работы была разработка системы удалённого управления учебным робототехническим комплексом (УРТК).

Система управления должна реализовывать следующую функциональность:

- установление соединения между клиентом и сервером;
- передача команд о запуске движения по определённой координате и в определённом направлении;
- передача команд об остановке движения по определённой координате;
- передача координат с сервера клиенту.

Система удалённого управления состоит из двух частей: *серверной* и *клиентской*. Клиентская и серверная части написаны на языке программирования Java; клиентская часть — на платформе Java 2 Micro Edition, а серверная — на Java 2 Standard Edition.

Серверная часть устанавливается на персональном компьютере с любой ОС (Windows, Linux) с установленной Java Virtual Machine. К компьютеру через LPT-порт подключён УРТК.

Клиентская часть устанавливается на мобильном телефоне. Мобильный телефон должен иметь цветной дисплей, желательно с разрешением не менее 176 на 132 пиксела, иметь подключение к сети Интернет (чаще всего используется GPRS).

Серверная часть позволяет управлять УРТК с локального компьютера или удалённо. Для выбора режима работы служат переключатели «*Управление с ПК*» и «*Удалённое управл.*» в правой части окна программы. Для управления с локального компьютера предназначены десять кнопок. Символ «*S*» (Stop) на кнопке означает, что в данный момент движение по данной координате в данном направлении остановлено. Символ «*R*» (Run) означает, что в данный момент движение по данной координате в данном направлении запущено. Также на экране отображаются текущие значения параметров.

Для удалённого управления необходимо переключится в режим «*Удалённое управл.*», ввести номер порта, который будет прослушиваться программой, и нажать кнопку «*Прослушивать*», после чего

программа переходит в режим ожидания подключения клиента к указанному порту. Вся информация о подключении клиента, командах старта и остановки движения, запрашиваемых координатах будет отображаться на панели «*Диалог Клиент-Сервер*». Одновременное управление с локального компьютера и удалённо невозможно.

Клиентская часть устанавливается на мобильном телефоне. Перед подключением к серверу необходимо в окне «*Настройки*» указать IP-адрес и порт сервера. Передача данных между клиентом и сервером осуществляется по протоколу TCP/IP. При переходе в окно управления происходит подключение к серверу.

При этом клиент создаёт два подключения к серверу. Одно для передачи команд и получения подтверждений об их выполнении, второе подключение используется для запроса координат.

Для управления служат десять экранных кнопок. Перемещение между кнопками происходит нажатием клавиш «2», «4», «6» и «8» или с помощью джойстика мобильного телефона. Нажатие экранной клавиши осуществляется с помощью клавиши «5» или нажатием на джойстик. Переход в предыдущее окно осуществляется нажатием клавиши «0».

После нажатия, на какую либо экранную кнопку серверу передаётся команда о старте или остановке движения по данной координате в данном направлении. При запуске движения цвет кнопки меняется на красный.

Обмен сообщениями между клиентом и сервером регламентируется протоколом прикладного уровня, который определят синтаксис и семантику команд. Команды можно разделить на три группы:

Сообщение приветствия передаётся при подключении клиента к серверу и информирует о том, что сервер готов к дальнейшей работе с клиентом.

Сообщения о старте или остановке движения передаются при нажатии на одну из экранных кнопок и имеют следующий формат. Первый символ определяет действие — запустить или остановить («*r*» или «*s*»), второй символ определяет координату («*x*», «*y*», «*z*», «*f*» или «*w*») и третий символ определяет направление движения («*r*» или «l»). Например, команда *rxl* сообщает серверу о необходимости запуска движения по координате «*x*» влево. Направление вправо направлено на увеличение координаты, направление влево направлено на уменьшение координаты.

В случае успешного выполнения команды сервер отвечает сообщением «*[выполняемая команда]* `ok`»; в случае ошибки при выполнении команды передаётся сообщение об ошибке «`error!`».

Команда «*k*» клиента является запросом текущих значений координат. В ответ сервер отправляет строку определённого формата, в которой содержаться координаты, их значения и флаги срабатывания концевых датчиков. Например, в строке `x50.23;y56.24;z50.0;wmax100;fmin0;` передаются значения всех координат (флаги «*max*» и «*min*» означают, что сработали концевые датчики по координатам «*w*» и «*f*»). Координаты запрашиваются 2 раза в секунду. Передаются только значения тех координат, которые изменились с момента предыдущего запроса координат.

Разработанная система удалённого управления соответствует всем поставленным требованиям: установление соединения между клиентом и сервером, передача команд, получение клиентом координат с сервера. Система управления является легко модифицируемой для управления другими объектами. Одним из преимуществ данной системы является то, что клиентская часть устанавливается на обычный мобильный телефон, что позволяет иметь доступ к управлению объектом из любой точки мира, находящейся в зоне обслуживания мобильного оператора. Удалённое управление возможно и с удалённого компьютера при наличии установленного эмулятора из программного продукта Sun Java Wireless Toolkit.

Список литературы

1. Евсяков А. С., Багаев Д. В. Система удалённого управления учебным робототехническим комплексом. — М.: ВНТИЦ, 2008. — №50200801833.
2. Dmitry Bagayev, Artem Evsyakov Portable system control of the educational robotic complex // The Second International Conference «Problems of Cybernetics and Informatics» (PCI' 2008), September 10–12, 2008, Baku, Azerbaijan, 2008. — p. 133–136.

Михаил Гусаров Новосибирск, ALT Linux

Опыт управления открытым проектом OpenInkpot

Рамки проекта и их влияние на развитие

Открытые проекты, не имеющие финансовой поддержки, зависят в своём существовании от притока добровольцев-разработчиков для восполнения естественной убыли по разным причинам членов проекта.

Наложение ограничений на участие в проекте (обладание малораспространённым устройством, знание языка, помимо английского, обилие недокументированной информации об устройстве и процессах проекта) неизбежно ведёт к уменьшению желающих работать над проектом и может поставить его на грань выживания.

Управление качеством

Задание критериев качества при старте проекта позволяет избежать или сильно ослабить проблемы с качеством в дальнейшем. Отсутствие таких критериев вместе с отсутствием механизма peer review не позволяет как оценивать качество, так и гарантировать какой-либо его уровень.

Выработка критериев качества во время жизни проекта гораздо более болезненна, чем в начале, и сопровождается долгими спорами участников. Единственная работающая практика исправления запущенной ситуации — «делай, как я».

Взаимодействие с коммерческими заказчиками

Коммерческие заказчики, нанимающие членов команды или спонсирующие разработку временем своих сотрудников, сильно меняют ландшафт проекта.

Положительные изменения:

- Работа над небольшими и средними проектами значительно ускоряется. В больших проектах такая помощь чувствуется меньше;

- В случае найма ключевых членов команды проекта повышается качество разработки за счёт увеличения вклада ведущих разработчиков в проект;
- Заказчик спонсирует разработку скучных или непрестижных частей, за разработку которых никто из добровольцев не брался.

Опасности:

- При неразвитой практике peer review возможно ухудшение качества для достижения краткосрочных коммерческих целей.
- Заказчик, нанимающий большую часть членов команды, получает возможность диктовать курс развития проекта, а также опосредованно удушает все разработки, помимо той, которая ему интересна. Кроме того, в этом случае может распасться и уже сложившийся механизм peer review.

Евгений Чичкарёв Мариуполь, Украина, Приазовский государственный технический университет

Сервер вычислений и web-интерфейс для работы с математическими и статистическими пакетами

Аннотация

В докладе представлен анализ существующих решений по разработке web-интерфейса к вычислительным пакетам — Sage, Octave, R, Maxima и др., проанализированы достоинства и недостатки аналогичных проприетарных решений для Maple, Matlab, MathCad. Представлены результаты разработки и опробования вычислительного сервера для проведения лабораторного практикума по различным учебным дисциплинам, основанного на открытом программном обеспечении (Octave, Scilab, R, Maxima).

В настоящее время для работы с вычислительными пакетами широко используются различные варианты клиент-серверных технологий с доступом пользователей к вычислительному ядру того или иного пакета через web-интерфейс. В этом случае на клиентской стороне необходим лишь интернет-броузер (обычно с поддержкой JavaScript), а вся вычислительная работа выполняется на серверной стороне.

Достаточно характерный пример — Matlab Web server (MWS), позволяющий организовать удалённую работу с MatLab. Скрипты

MatLab, предназначенные для удалённого запуска с использованием MWS, требуют некоторой доработки, заключающейся в создании web-интерфейса (в простейшем варианте считываются данные из полей html-формы). Другой характерный пример — Mathcad Calculation Server, позволяющий работать с MathCad-документами в сети. При подготовке документа к публикации пользователь вводит исходные данные расчётов в текстовые поля и другие элементы интерфейса (WebControls), передаёт их на сервер, где производятся вычисления, и получает результаты вычислений, в том числе и в графической форме.

Аналогичные расширения существуют и для Maple (MapleNet) или Mathematica (webMathematica).

Таким образом, все проприетарные программные средства удалённой работы с вычислительными пакетами требуют некоторой доработки скриптов или документов для организации web-интерфейса, а также имеют встроенные механизмы упрощения его создания. Технологически средства организации удалённого доступа несколько различаются, но характерным примером может быть webMathematica, основанная на технологиях сервлетов и JSP.

Что касается вычислительных пакетов с открытым кодом, то наиболее развитые средства для организации удалённой работы существуют для R. Наиболее развитый R-пакет, позволяющий запустить TCP/IP-сервер для доступа других программ к возможностям R, — это Rserve (`http://www.rforge.net/Rserve/`). Данный пакет обладает достаточно широкими возможностями как для организации удалённой работы, так и для взаимодействия с R через localhost (в частности, из OpenOffice). Наряду с Rserve, возможно и низкоуровневое взаимодействие java-R-java при помощи пакетов JRI и rJava (`http://www.rforge.net/JRI`, `http://www.rforge.net/Rserve/` и `http://www.rforge.net/rJava`). Наряду с достаточно сложным Rserve, существует и более простой в использовании Rphp (`http://dssm.unipa.it/R-php/`), позволяющий организовать удалённую работу с R при помощи web-сервера Apache и набора скриптов на php. Существуют и другие варианты web-интерфейса R (см. FAQ на `http://cran.r-project.org`).

Пакеты, предназначенные для удалённой работы с Octave и Maxima (см. `http://hara.mimuw.edu.pl/weboctave/web/` и `http://maximaphp.sourceforge.net/`), по организации и возможностям мало отличаются от Rphp.

При запуске вычислительного ядра того или иного пакета из php (при помощи функций *passthru* или *shell_ exec*) важной проблемой является фильтрация команд пользователя для блокирования потенциально опасных функций или команд конкретного пакета, требующих доступа к файловой системе сервера.

Пакеты, написанные на php, обычно включают и функции системы управления контентом: регистрацию пользователей, хранение, редактирование и удаление скриптов пользователей и т. п.

Возможности удалённой работы с вычислительными пакетами обеспечивает и пакет Sage (`http://sagemath.org/`). При помощи Sage-notebook можно удалённо работать с целым рядом математических пакетов, входящих в состав Sage, а также публиковать в Интернете интерактивные документы («склеивая» скрипты для различных пакетов и результаты их работы при помощи кода на Python).

Как показало опробование различных вариантов организации удалённого доступа к вычислительным пакетам, для организации студенческого практикума и дипломного проектирования наиболее простым и удобным решением (при условии работы внутри локальной сети) оказалась разработка скриптов доступа на php и JavaScript и разработка html-форм для различных задач.

Веб-сервер Apache был развёрнут под управлением ОС Linux Suse 11.1. Предусмотрена работа с пакетами Octave, R, Maxima (с возможностью расширения круга используемых вычислительных приложений). Для каждого пакета разработан (на базе Rphp) комплекс скриптов для интерактивного выполнения команд, а также выполнение файлов конкретного пользователя (после регистрации и входа в систему под своим логином, при этом создаётся каталог данного пользователя) либо обращения к стандартным моделям и сценариям, обеспеченным формами и протоколами для ввода исходных данных и интерпретации результатов расчёта. Для хранения базы данных аутентификации использована MySQL.

Достоинством данной разработки являются широкие возможности настройки на конкретную задачу. В частности, с её использованием были разработаны система статистического анализа результатов поплавочного контроля в чёрной металлургии (R), система статистического анализа и прогноза временных рядов (Maxima), пакет моделирования динамических систем с распределёнными и сосредоточенными параметрами (Maxima, Octave) и др.

Юрий Бушмелев Ульяновск, ИП Бушмелев Юрий Юрьевич
Проект: OpenEmbedded http://www.openembedded.org

OpenEmbedded — система сборки дистрибутивов Linux для встраиваемых и мобильных устройств

Аннотация

OpenEmbedded — это фреймворк, позволяющий создавать собственные дистрибутивы Linux для встраиваемых и мобильных устройств. OpenEmbedded сочетает в себе систему сборки с множеством готовых рецептов и богатые возможности по описанию новых дистрибутивов и новых аппаратных платформ. На данный момент в репозитории зарегистрировано более 30 дистрибутивов, имеется поддержка более 200 устройств, а также существует более семи тысяч «рецептов» для сборки приложений.

Проект OpenEmbedded был создан в качестве замены системы сборки проекта OpenZaurus, базировавшейся на buildroot. За основу при разработке была взята система портежей Gentoo и утилита emerge.

Основными целями проекта являлись:

- возможность кросс-компиляции;
- поддержка зависимостей между пакетами;
- возможность генерировать бинарные пакеты (tar, rpm, deb, ipk);
- возможность создавать образы и репозитории из бинарных пакетов;
- поддержка множества архитектур, машин и дистрибутивов;
- простота описания метаданных и возможность их повторного использования.

В результате, был создан набор метаданных (рецептов), используемых для кросс-компиляции, упаковки и установки ПО. Для управления задачами сборки ПО и отслеживанием зависимостей была разработана утилита BitBake, которая позже выделилась в отдельный проект.

На данный момент в основном репозитории OpenEmbedded зарегистрировано более 30 дистрибутивов, наиболее известными из которых являются Ångström и KaeliOS. Некоторые дистрибутивы также

используют собственные бранчи (dreambox, openzaurus) и репозитории (openmoko, poky). Кроме «полноценных» дистрибутивов, также имеется возможность собирать пакеты для других дистрибутивов, не базирующихся на OpenEmbedded (OpenWRT, maemo).

«Дистрибутив» в рамках OpenEmbedded — это файл с описанием версий ПО и ядра для различных устройств и списком задач, которые необходимо выполнить для получения образа дистрибутива. Для отдельных устройств можно задать отдельные версии ПО и назначить отдельные дополнительные задачи.

Сейчас OpenEmbedded поддерживает более 200 устройств. Это как реальные устройства (например линейка КПК Sharp Zaurus, платы BeagleBoard, Atmel AT91* и др.), так и виртуальные машины (vmware, qemu в режиме эмуляции x86 и arm). В описании устройства можно задать, в частности, архитектуру процессора (i486/i568/i686, arm, mipsel, avr32, powerpc, sparc и т. д.), тип образа ядра (bzImage/uImage), версию ядра, наличие периферии (wifi, bluetooth, экран, клавиатура, usb host и т. д.).

Само ПО и типовые задачи представлены в виде «рецептов» (recipes). Это файлы, в которых описано, откуда брать ПО, какие ему нужны зависимости, как его компилировать, как устанавливать и в какие пакеты собирать. Можно выставлять различные параметры в зависимости от дистрибутива и целевого устройства. Также поддерживаются локальные оверлеи.

Управляет метаданными утилита BitBake. Она собирает всю информацию о конфигурации (настройки дистрибутива, устройства и пользователя) воедино, строит граф задач с обходом зависимостей, а затем последовательно выполняет все задачи. Как правило, в первую очередь создаётся окружение для кросс-компиляции, в котором производится дальнейшая компиляция и сборка пакетов. Обработка типового «рецепта» включает в себя следующие задачи: загрузку исходных кодов, распаковку, применение патчей, компиляцию, установку и создание пакета или образа.

Благодаря своей гибкости OpenEmbedded успешно применяется для построения коммерческих решений. Следующие продукты были разработаны на базе OpenEmbedded:

- Palm WebOS — ПО и SDK для коммуникатора Palm Pre;
- ПО для нетбука Touch Book компании Always Innovating;
- ПО для телефона Neo FreeRunner (GTA02) компании OpenMoko;
- ПО для STB Dreambox 702x.

Специально для повышения привлекательности у производителей коммерческих решений в начале года был создан стабильный бранч (stable/2009), в рамках которого производится стабилизация метаданных. Это, в частности, позволило компании Bug Labs перевести свои разработки с репозитория Poky на работу с OpenEmbedded напрямую.

К сожалению, высокая гибкость системы является одновременно и её недостатком. Система довольно сложна в изучении, а парсинг метаданных занимает довольно много времени.

Тем не менее, учитывая растущий интерес к встраиваемым и мобильным устройствам под управлением Linux, следует прогнозировать рост популярности OpenEmbedded, в том числе и среди производителей коммерческих решений. Будучи однажды освоенной, OpenEmbedded позволит быстрее выводить на рынок новые решения, что в настоящее время является серьёзным конкурентным преимуществом.

Евгений Сыромятников, Георгий Курячий Москва, ALT Linux
Проект: семинар UNИХ http://uneex.ru/

Использование wiki-сервера MoinMoin для совместной разработки документации

Аннотация

В докладе освещается проблема совместной разработки документации, рассматривается возможность использования wiki-технологий для решения задач, возникающих в рамках данной проблемы и предлагается решение на базе модифицированного wiki-сервера MoinMoin, позволяющее организовать процесс совместной разработки документации.

Предпосылки

Летом 2008 года был начат проект по документированию ПСПО [1], [2]. В его рамках требовалось создание комплекта документации, удовлетворяющего ряду требований. К особенностям проекта можно

причислить жёсткие временные рамки, возникшие по различным причинам. В результате анализа требований было решено использовать для совместной разработки wiki-сервер `MoinMoin` с необходимыми модификациями. Далее рассматривается полученный набор модификаций.

Постановка задачи

Рассмотрим условия и допущения, в рамках которых мы рассматриваем задачу совместной разработки документации:

- Документация представляет собой набор документов, связанных гипертекстовыми ссылками.
- Структура документации может быть заранее не определена.
- Документация организована в виде курсов, состоящих из модулей. Данный подход позволяет разрабатывать отдельные части документации независимо.
- Форматы материалов, используемых в документации, не ограничиваются, при этом процесс работы с ними должен быть по возможности унифицирован.
- Процесс разработки централизован. Анализ случая использования децентрализованных средств разработки выходит за рамки данной работы.
- Мобильность опубликованной версии. ПО, позволяющее просматривать опубликованную документацию, должно быть переносимым — не иметь или иметь минимальное количество системных зависимостей.

Wiki и MoinMoin

Выбор wiki в качестве технологии совместной разработки был сделан по следующим причинам:

- Wiki изначально ориентирована на совместную работу, и её реализации предусматривают соответствующие инструменты (контроль версий, прав доступа, механизм разрешения конфликтов).
- Простота: документ в wiki-разметке представляет из себя текстовый файл.
- Гибкость: набор wiki-документов не имеет предопределённой жёстко заданной структуры.

Выбор MoinMoin в качестве основы был обусловлен рядом факторов:

- Современная wiki-платформа, активные сообщество и поддержка.
- Написан на Python, легко расширяется и модифицируется.
- Имеет минимальные требования по установке в плане программных зависимостей (на сервере требуется только наличие интерпретатора Python).

Реализация

Рассмотрим подробнее возможности, предлагаемые MoinMoin, которые можно использовать для организации сместной разработки документации.

- Механизм парсеров и генераторов (`parsers`, `formatters`) с использованием промежуточного формата на основе wikiDOM [3] позволяет использовать большое количество входных и выходных форматов. В рамках проекта документирования ПСПО на основе этих механизмов был реализован экспорт в HTML-tree, впоследствии был добавлен частичный экспорт в формат Moodle course backup.
- Использование фильтров-обработчиков, извлекающих текстовую информацию из файлов различных форматов, позволяет генерировать поисковый индекс и использовать его.
- Механизмы расширения функциональности MoinMoin (`actions`, `macros`) позволяют добавить возможности, необходимые для решения задачи, которые изначально отсутствовали в MoinMoin.

Возможности

Для единого способа обработки разнородных внешних материалов была выделена часть с метаинформацией (т. н. *паспорт*), в котором содержатся данные об авторе, источнике, лицензии, формате, а также краткая аннотация. Содержательная часть материала сохраняется либо в wiki, либо в статическом хранилище и, по возможности (в зависимости от наличия фильтра для данного формата), индексируется. Для работы с материалами добавлена возможность создания ссылок на них (на паспорта или на собственно файлы).

Для поддержки модульности были написаны макросы, обрабатывавшие таблицы специального вида на страницах модулей и в соответствии с информацией, указанной в них. Данные макросы позволяли отслеживать готовность как модулей, так и курсов.

Публикация. Здесь можно выделить следующие аспекты:

- Создание автономного комплекта документации;
- Публикация части документации, замкнутой по зависимостям.

Первая задача решалась путём модификации wiki-сервера MoinMoin (это требовалось в связи с тем, что он не был расчитан на работу с носителем, к которому нет доступа на запись). Вторая задача решалась построением графа зависимостей и выделением необходимого подграфа.

Результаты

Создан набор дополнений и модификаций wiki-сервера MoinMoin, успешно использовавшийся в рамках проекта документирования ПС-ПО [2]. Данные модификации опубликованы на сайте проекта [4] и распространяются в соответствии с GPL.

Вместо заключения

Серебряной пули нет [5]. Данное решение не позволяет делать документацию из чего угодно в произвольном выходном формате. Оно лишь может помочь организовать процесс сбора и категоризации имеющихся материалов таким образом, чтобы ими было чуть удобнее пользоваться.

Литература

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

[2] *Курячий Г. В., Сыромятников Е. Л.* Командная разработка свободных учебных материалов по проекту «Документирование пакета свободного программного обеспечения», *«Свободное программное беспечение в образовании. Сборник тру-*

дов Всероссийской конференции (г. Челябинск)», под редакцией А. В. Панюкова, Издательство ЮУрГУ, 2009, стр. 14—18. http://freeschool.altlinux.ru/wp-content/uploads/2009/03/chelyabinsk_25-26_march_2009.pdf.

[3] http://moinmo.in/MoinDev/WikiDom

[4] http://disc.uneex.ru/

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

Вадим Мутилин, Алексей Хорошилов Москва, Институт системного программирования РАН

Проект: Верификация Linux-драйверов
http://linuxtesting.ru/project/ldv

База правил для верификации драйверов Linux

Аннотация

Статический анализ программ, нацеленный на обнаружение ошибок, специфичных для целевого ПО, активно развивается. Но для полного использования его потенциала необходимо решить вопрос идентификации и формулировки правил, описывающих искомые ошибочные ситуации для заданной целевой системы. В докладе данная проблема рассматривается на примере формирования базы ограничений на взаимодействие между драйверами устройств и сердцевиной ядра Linux, которая позволит применять инструменты статического анализа для автоматической верификации драйверов ОС Linux.

Александр Котельников Санкт-Петербург, ALT Linux
Проект: Skytap Virtual Lab http://www.skytap.com

Виртуальные сети в Cloud Computing

Аннотация

Одной из отличительных черт Skytap Virtual Lab является предоставление сетевой инфраструктуры как части сервиса. Простая конфигурация сети по умолчанию и возможность в некоторых пределах изменить её позволяют вынести на виртуальные машины не только отдельные системы, но и их комплексы. Компоненты, виртуализация которых невозможна или нецелесообразна, могут быть подключены к виртуальной части как непосредственно через Интернет, так и с использованием VPN-решений.

Реализация, тестирование и развёртывание решения, предоставляющего пользователю возможность построить сеть виртульных машин и эффективно управлять как сетью в целом, так и сетевыми свойствами отдельных машин, а также тем, как виртуальные машины связаны с внешним миром, стало возможным благодаря использованию Linux в качестве маршрутизатора и операционной системы, в состав которой входят open source реализации таких сервисов, как dhcp, named, smb, без которых порой не мыслимо управление сетью.

Стоящая перед разработчиками задача состояла в том, чтобы без использования чрезмерных вычислительных ресурсов предоствить пользователям удобный интерфейс к виртуальным машинам, при том что основным инструментом для доступа к ним является веб-браузер.

Сервис включает в себя (но не ограничивается им) управление машинами как логическими единицами (создание, удаление, объединение в группы), как практически реальными компьютерами (запуск, выключение, приостановка, изменение конфигурации), доступ к терминалу виртуальной машины, возможность соединения как с внешними ресурсами из виртуальной машины, так и с виртуальной машиной из внешнего мира.

Практически все эти задачи, кроме непостредственно виртуализации, решены с помощью программного обеспечения с открытым кодом, а это, в частности, позволяет относительно дёшево создавать мини-копии инсталляций решения, что крайне удобно для независимой отладки и тестирования разных версий самого продукта. Де-

факто, у каждого разработчика имеется свой стек от гипервизора до веб-сервера.

Описание части продукта, предоставляющей сетевые сервисы в этой системе, является темой настоящего доклада.

Специфичность проблемы состоит в том, что один или разные пользователи могут создавать «одинаковые» сети, например, посредством создания копий уже существующих, при этом система, естественно, должна различать их и, за исключением тех случаев, когда это логически противоречиво, позволять им функционировать одновременно. Логические конфликты появляются при подключении ресурсов, к которым требуется предоставлять эксклюзивный или каким-то образом ограниченный доступ. При этом, как и в решении прочих техничеких проблем, хочется оградить пользователя от ощущения, что при переходе от физичеких машин к виртуальным на него наложили дополнительные ограничения.

Артём Маковецкий Москва, Мототелеком

Проект: Семейство продуктов IP-PBX Mototelecom
`http://www.mototelecom.ru/`

Развёртывание систем телефонии в современных условий

Аннотация

На данный момент времени задача обеспечения офиса телефонной связью выходит далеко за рамки прокладки телефонных проводов и наладки офисной мини-АТС. Высокая степень проникновение систем автоматизации различных уровней в малый и средний бизнес накладывает свои ограничения и требования к системам связи. Разварачиваемое решение должно быть интегрированно с используемыми на предприятии системами автоматизации, иметь простое управление и позволять автоматизировать большинство задач, решение которых связано с работой с системой связи. Идеальной платформой для создания таких решений могут служить разлчные дистрибутивы Linux.

Рассматривая структуру современного решения по телефонизации офиса, можно заметить, что собственно телефония не является единственной его частью. Сервис, предоставляемый телефонной частью

решения, вполне определён и понятен, но его явно не достаточно. На первый план выходит забота о дополнительных сервисах.

Одним из самых важных факторов систем связи становится простота и удобство управления. Гибридные АТС, которые программировались специально обученными людьми при помощи толстых справочников, безворотно ушли в прошлое. Современный интрефейс системы связи должен быть прост, понятен, а также доступен с рабочего места любого сотрудника компании. Самым естественным решением этой задачи является разработка web-интерфейса. А значит, система телефонии дополняется полноценным web-сервером и, естественно, сервером баз данных.

Далее стоит выделить, что раз система представляет собой сервер IP-телефонии, а также управляется посредством web-интерфеса, то она является полноправным участником офисной сети и требует соотвествующих настроек. Следовательно, в решение включается web-интерфейс управления сетевыми настройками, а также сетевой экран на основе iptables.

Еще одной важной задачей администрирования является резервное копирование (backup) работающей системы. Исходя из ориентации решения на компанию, в штате которой может не быть специалиста по Unix-системам, задача резервного копирования должна решаться наиболее простым (с точки зрения управления) способом. Хорошим вариантом является собственная система резервного хранения в паре со встроенным мастером восстановления системы. Ещё одна сугубо администраторская задача — контроль за состоянием жёстких дисков. И опять единственное решение — просто интерфейс настройки контроля и автоматизированное решение данной проблемы. Все описанные плановые задачи требуют активного использования планировщика заданий, следовательно, cron становится неотемлимой частью решения телефонии.

Кроме сервисных задач, необходимо решать задачи, так или иначе связанные непосредственно с телефонией. Одна из них — работа с факсовыми документами. Система должна содержать в себе набор графических утилит, которые позволяют конвертировать принятые факсы в другие форматы, например PDF, как наиболее удобный в использовании, а также производить минимальное редактирование принятого документа. Задачи конвертации решаются и при работе с записанными разговорами, т. к. требуется экономить место на жёстком

диске и хранить разговоры в сжатом виде. Следовательно, в решение следует включить набор утилит для работы с графикой и звуком.

Продолжая тему работы системы с различными файлами, следует обозначить, что пользователям придётся активно «скачивать» и загружать их на сервер телефонии. Для работы с хранилищем телефонных разговоров удобно использовать ftp-доступ, особенно когда требуется скачать/удалить большое число голосовых записей. Но ftp — не единственный протокол, который используется для файлообмена между пользователем и системой. Электронная отправка факса осуществляется при помощи сохранения факсового документа в определённой сетевой директории. Поддержка такого функционала требует включения в состав решения Samba-сервера.

Подводя итог, можно отметить, что в данный момент значительную часть системы телефонии занимают вспомогательные сервисы. Наиболее удобной платформой для этого являются Unix-системы, которые облодают большим репозиторием прикладных программ, а сами приложения, в большинстве своём, обладают открытым кодом, а значит в большей степени удобны для интеграции.

Андрей Михеев Москва, Консалтинговая группа Руна
Проект: Runa WFE http://sourceforge.net/projects/runawfe

Автоматизация документооборота предприятия при помощи открытой системы управления бизнес-процессами предприятия Runa WFE

Аннотация

В докладе рассказывается, как при помощи интеграции системы Runa WFE с одной из открытых ECM-систем можно автоматизировать документооборот предприятия.

Описание системы Runa WFE

RUNA WFE — открытая, масштабируемая, ориентированная на конечного пользователя система управления бизнес-процессами пред-

приятия. Система платформонезависима (написана на Java), распространяется под LGPL-лицензией.

Основная задача системы — раздавать задания исполнителям. Последовательность заданий определяется графом бизнес-процесса, который менеджер или бизнес-аналитик может быстро изменять при помощи редактора бизнес-процессов.

Элементы системы:

- BPM-система;
- Графический редактор процессов;
- Клиент-оповещатель о поступивших заданиях.

BPM-система:

- Работа с определениями и экземплярами процессов;
- Работа со списками заданий;
- Визуализация форм, соответствующих заданиям;
- Работа с системой через веб-браузер;
- Предоставление возможности работы с системой приложениям специального вида (ботам[1]);
- Авторизация и аутентификация пользователей.

Графический редактор процессов:

- Редактирование графа процесса;
- Создание и редактирование графических форм заданий;
- Создание и назначение ролей;
- Создание переменных.

Клиент-оповещатель о поступивших заданиях:

- Оповещение о поступивших заданиях;
- Работа с системой через специальное приложение-клиент.

Система является как бы конвейером, перенесённым с производства в офис, позволяет работнику выполнять поступившие задачи, не отвлекаясь на:

- Получение необходимой для выполнения задания информации;
- Передачу результатов своего труда другим работникам;
- Изучение должностных инструкций.

[1]В частности, боты могут моделировать работу сотрудника предприятия

Всё необходимое возникает на экране пользователя при «клике» на задании (в частности, на экране может быть написана инструкция — как следует выполнять полученное задание)

Исполнителями могут быть как люди, так и специальные компьютерные приложения — боты. В частности, используя ботов, можно решить задачу автоматической генерации документов.

Организация документооборота при помощи интеграции Runa WFE и одной из свободных ECM-систем

При помощи переменных в системе управления бизнес-процессами происходит передача информации между исполнителями заданий. Если в переменных бизнес-процесса хранить документы, то систему можно использовать для перемещения документов от исполнителя к исполнителю.

Однако системы управления бизнес-процессами хорошо решают задачи управления, но не предназначены для хранения документов, контроля версий документов и поиска документов по их атрибутам.

Поэтому предлагается автоматизировать документооборот при помощи интеграции системы Runa WFE с одной из открытых ECM-систем. В качестве таких систем можено использовать, например, Alfresco, Nuxeo, или Jackrabbit. При этом в Runa WFE предлагается производить графическое моделирование, осуществлять перемещение точек управления бизнес-процессами и обмен данными, автоматически формировать документы по шаблонам.

В определённых точках бизнес-процессов боты будут отгружать документы в ECM-систему, а уже в ней предлагается автоматизировать хранение документов, ведение их версий, а также поиск документов.

Во время доклада предполагается продемонстрировать примеры решений по реализации документооборота.

Литература и ссылки

1. Он-лайн демо-версия системы Runa WFE доступна по адресу: `http://wf.runa.ru/Online_Demo`
2. Ссылка на проект Runa WFE: `http://sourceforge.net/projects/runawfe`

3. Ссылки на открытые ECM-системы:
   - Alfresco — `http://www.alfresco.com`
   - Nuxeo — `http://www.nuxeo.com`
   - Jackrabbit — `http://jackrabbit.apache.org`

Александра Панюкова Москва, ALT Linux
Проект: ALT Linux Children `http://www.altlinux.ru`

Дистрибутив ALT Linux Children: опыт и перспективы

Аннотация

Специализированный дистрибутив для детского творчества ALT Linux Children успешно развивается и применяется уже два года. В докладе рассматриваются технические вопросы формирования такого дистрибутива, обосновывается выбор ПО, описываются существующие варианты дистрибутива и перспективы его развития.

ALT Linux Children

Программное наполнение ALT Linux Children составлялось с расчётом не только на непосредственное проведение курса, но и на последующую самостоятельную работу детей дома. В дополнение к программным продуктам, необходимым для самого курса, в дистрибутив входят инструменты для более требовательных или любознательных пользователей: диспетчер фотографий цифровой фотокамеры, редакторы фрактальной и ASCII-графики, мощный аудиопроигрыватель, программы по астрономии и географии. По сравнению с предыдущей версией дистрибутива, значительно обновлены графический редактор GIMP и редактор нелинейного видео Kdenlive.

Для дошкольников от 4 лет и младших школьников предусмотрен развивающий центр GCompris, который содержит множество модулей, начиная с освоения клавиатуры и мыши и заканчивая логическими играми и заданиями, помогающими изучать окружающий мир. Для цели общей тренировки добавлен клавиатурный тренажёр.

Стоит отдельно отметить, что в состав дистрибутива не входят офисные программы, почтовые клиенты, средства манипуляции контактной информацией и прочие продукты, не отвечающие поставленным задачам. Практика преподавания показала, что эти программы создают значительный отвлекающий фон и засоряют информационное пространство при проведении занятий. По той же причине подключение к сети Интернет переведено в «ручной» режим и по умолчанию не используется.

Перспективы и пути развития

Live CD

По окончании работ по курсу для детей [2] планируется релиз Live CD с выпуском книги по курсу.

Кроме того, планируется выпуск Live DVD с более подробным набором исходных материалов (в том числе с видео), большим количеством игрушек и, конечно, возможностью настроить Интернет, об отсутствии которой очень сожалеют родители.

Возможно наличие «профилей загрузки» на Live CD: сейчас по умолчанию после загрузки ребёнок получает систему, в которой все носители информации примонтированы на запись. Такая схема идеальна для того, чтобы ребёнок мог сохранять созданные файлы на любых носителях, чтобы не потерять их к моменту завершения работы. Однако не все родители уверены в том, что, обладая правом записи на все жёсткие диски, ребёнок не повредит информацию. Поэтому рассматривается возможность создания нескольких вариантов загрузки:

- Со всеми носителями, доступными на запись.
- С жёсткими дисками, доступными только на чтение. При этом все съёмные носители доступны на запись.
- Только съёмные носители доступны на запись.

Установочный вариант

Домашний для детей

Есть некий класс детей, которые утверждают, что компьютер в доме используется только ими, что они отдают отчёт в своих действиях и т. п. Есть некий класс родителей, которые были бы не против, чтобы

ALT Linux Children был установлен на компьютере, не было бы необходимости постоянно загружаться с Live CD. Логично предположить, что такой вариант дистрибутива тоже имеет право на существование. С другой стороны, такое решение не может быть универсальным, если компьютер в доме используется не только и не столько ребёнком, сколько родителями. Устанавливать отдельный дистрибутив для ребёнка довольно неудобно: разница между загрузкой с Live CD и установленной второй системой (особенно если один Linux уже стоит) с точки зрения удобства процесса работы невелика. Устанавливать для ребёнка Children, а затем доустанавливать недостающие родителям программы тоже довольно неудобно. Можно привести учётную запись ребёнка-пользователя дистрибутива Desktop к состоянию, близкому к Children, но по количеству усилий этот вариант эквивалентен предыдущему.

Очевидно, что отдельный дистрибутив довольно неудобен с точки зрения конечного пользователя, как и любое размножение сущностей. В качестве варианта решения можно предложить вариант создания при установке более одного пользователя, но с разными профилями. Так, например, в случае Children, можно было бы по выбору устанавливающего создать дополнительного пользователя, отличного от основного некоторыми настройками, с доустановкой недостающих пакетов.

В качестве таких «небольших» настроек могут быть, например, ссылки с рабочего стола на некоторые ресурсы, специфичные для Children, может быть несколько изменена структура меню (специфичные для Children пункты выведены на передний план, а те, которые вряд ли пригодятся детям — наоборот) и т. п.

«Боекомплект преподавателя»

Другой вариант «Устанавливаемого Children», который может пригодиться — это установочные диски для быстрого разворачивания класса на неизвестной территории. Как показала практика, изначальный вариант с разворачиванием всего с чистого листа может применяться в практически любых условиях. Проблема, с которой можно столкнуться, очень похожа на проблему, известную по школьному проекту: железо везде разное, для разного железа оптимально подходят разные решения. Одно везде примерно одинаково: есть некий класс с примерно равными по характеристикам компьютерами. Воз-

можно, есть какая-то сеть (а возможно, и нет, или её можно организовать). В этом случае довольно универсальный вариант — установка на все компьютеры, при этом при наличии сети (или возможности её сделать) один компьютер выделяется под «сервер». Под «сервером» в данном случае понимается некий компьютер, не обязательно отличимый от других (один из класса, возможно, самый мощный), за которым будет работать ребёнок (или преподаватель, если будет возможность не использовать его детьми), но который будет отличаться от всех остальных только тем, что на нём будет работать некоторое количество служб, а также будут храниться все работы всех детей (для того чтобы у них была возможность работать за разными компьютерами, а не «драться» за место).

В качестве комплекта для установки предполагается 2 CD или 1 DVD, с которых будут устанавливаться и «сервер», и клиентские системы.

В первую очередь будет установливаться «сервер». Когда становится известен IP-адрес или имя узла (hostname) сервера, производится установка системы на остальные компьютеры, при установке прописываются необходимые значения с учётом установки сервера (чтобы не нужно было производить последующую настройку).

Ситуация с сетью может оказаться довольно противоречивой. Хорошо, когда сети нет вообще, либо когда сеть в классе более-менее обособлена. Однако это не всегда возможно. Поэтому необходим шаг, аналогичный выбору внутреннего/внешнего интерфейсов, например, в Школьном Сервере. Даже если сетевая карта одна, эта настройка применяется, в частности, для выбора, запускать dhcp-сервер или нет. Следует не забывать, что разрабатываемое решение готовится, может быть, и для подготовленного человека, но ограниченного по времени, поэтому, чем меньше различных настроек надо будет корректировать после установки в неизвестной среде — тем лучше. Экономия времени должна быть во всём: и в раскладывании (или генерации и раскладывании) ключей по клиентским компьютерам, и в управлении классом.

Итого, после установки с минимумом действий предполагается получать класс, в котором:

- будет один samba-сервер. На клиентских компьютерах на рабочем столе будет ссылка на соответствующий ресурс, куда дети смогут соханять свои работы. Исходные материалы также будут раздаваться по Samba, однако на всякий случай на каждом

компьютере будет локальная копия без ссылки с рабочего стола (если не был выбран бессетевой вариант установки).

- будет настроен jabber-сервер. На клиентских компьютерах — установленные и настроенные jabber- клиенты.
- возможно управление всеми компьютерами (в т. ч. и сервером) одновременно с любого компьютера класса (правда, пока только через командную строку).

К сожалению, надеяться на то, что установленные системы будут единственными на компьютерах, не приходится. Кроме того, нередки задачи, когда работы детей могут понадобиться на компьютере с Windows (например, поделиться с соседним классом, работающим под Windows, скопировать файлы на ноутбук с Windows, подключённый к сети и т. п.).

Литература

[1] *Панюкова А. А.* Создание обучающего курса для детей на базе Linux // Третья конференция «Свободное программное обеспечение в высшей школе», Переславль, 2–3 февраля 2008 г. — М., ALT Linux, 2008

[2] *Панюкова А. А.*, *Якшин М. М.*, *Панюкова Т. А.* Методика проведения учебных занятий с использованием свободно распространяемого программного обеспечения // Роль и место самостоятельной работы студентов в образовательном процессе вуза. Юбилейная региональная научно-методическая конференция (4–6 февраля 2008 г.): Сб. науч. тр. — Челябинск, Издательство ЮУрГУ: 2008. — Т. 1. — ISBN 978-5-696-03714-1

[3] *A. Panyukova*, *M. Yakshin*, *T. Panyukova*. Organization and methodics for realization of computer graphics studying using free software // Proceedings of the 10th International Workshop on Computer Science and Information Technologies, Antalya, Turkey, September 15–17, 2008: Сб. науч. тр — Уфа, Редакционно-издательский комплекс УГАТУ: 2008. — Т. 1. — С. 238 — ISBN 978-5-86911-787-8

Дмитрий Сподарец Одесса, Украина

Платформа TERM как средство для автоматизации измерительного оборудования

Разработка перспективных технологий требует применения современного диагностического и измерительного оборудования. Данное оборудование можно купить или получить, модифицируя имеющееся. Зачастую отработанная база методик и наличие старого оборудования для воплощения данных методик являются лучшим вариантом, чем приобретение готового оборудования. Основные преимущества этого варианта — возможность адаптации оборудования под конкретный опыт и экономия средств на закупке нового оборудования.

На кафедре теплофизики Одесского национального университета им. И. И. Мечникова был начат проект по созданию аппаратно-программного комплекса, позволяющего облегчить процесс модернизации имеющегося оборудования и разработку нового. Проект получил название «Платформа TERM» и выполнялся последовательно в рамках нашей квалификационной, бакалаврской и дипломной работ. Осуществлено три этапа, в результате чего реализованы следующие возможности Платформы:

- Этап первый: сбор информации и начало проектирования платформы; разработка модуля сопряжения термопара-компьютер.
- Этап второй: механизм диагностики температур при помощи различных термопар; механизм диагностики температуры пламени методом относительной яркостной пирометрии.
- Этап третий: механизм диагностики низкотемпературной плазмы спектральным методом.

В разработке активно принимали участие сотрудники кафедры, в частности профессора Драган Г. С., Калинчак В. В. и ст. преподаватель Черненко А. С.

Вся платформа разрабатывается на основе открытого/свободного ПО.

Основной принцип работы модуля сопряжения термопара-компьютер состоит в том, что первоначально напряжение с термопары усиливается операционным усилителем, а затем передаётся на

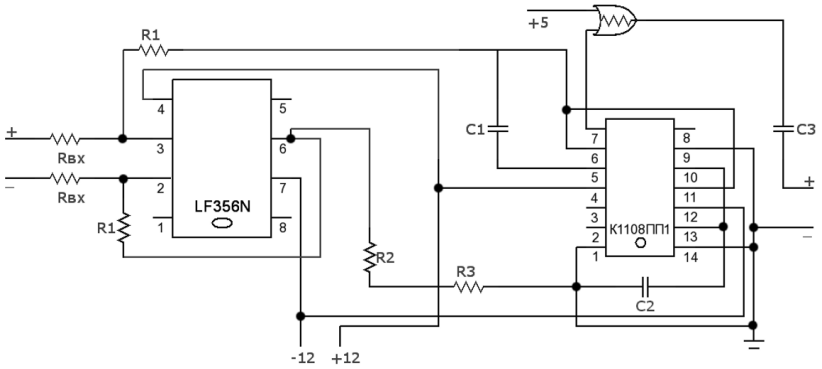

Рис. 1: Схема модуля сопряжения

преобразователь напряжение-частота. Далее импульсный сигнал подаётся на вход звуковой карты, с которой происходит считывание и подсчёт импульсов. По градуировке (соотношение количества импульсов/температура) определяется действительная температура. Схема модуля показана на рис. 1.

В основе механизма определения температуры пламени заложен метод относительной яркостной пирометрии. На основе полученного изображения пламени с цифровой видеокамеры определяется интенсивность его излучения, которая сравнивается с градуировочными данными. В результате получаем действительную температуру пламени.

В рамках механизма диагностики низкотемпературной плазмы реализована возможность определения температуры конденсированной и газовой фаз, а также концентрации электронов. Определение данных параметров происходит за счёт анализа спектра излучения плазмы. Анализ сплошного спектра позволяет найти температуру конденсированной фазы. Отношение интенсивностей спектральных линий позволяет найти температуру газовой фазы. А штарковская полуширина спектральных линий позволяет определить концентрацию электронов в плазме.

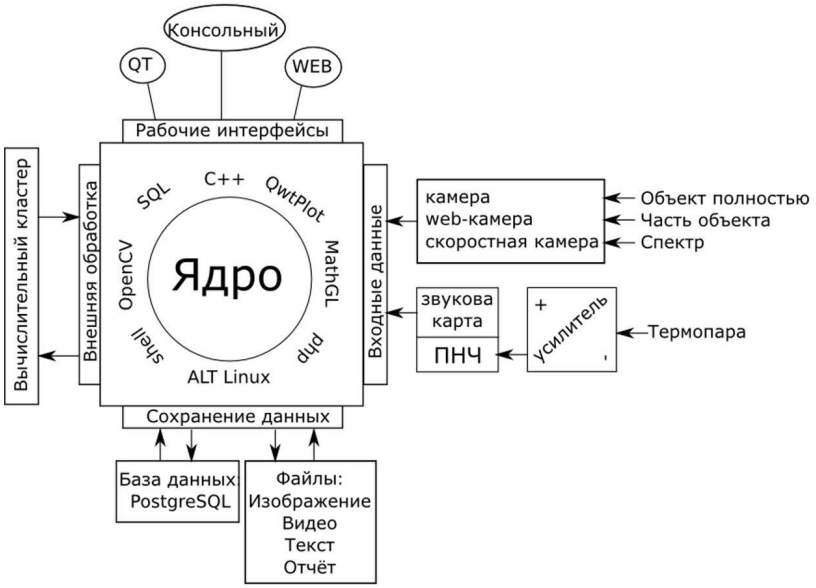

Рис. 2: Блок-схема комплекса

Есть три рабочих интерфейса: QT, Web и консольный. Сохранение данных происходит в базе данных, хотя их можно сохранять в различного рода файлах (текстовых, видео, изображениях и отчётах).

Для выполнения сложных вычислений разработана возможность проведения их на внешних устройствах, например на вычислительном кластере.

Платформа может работать в двух режимах: в режиме реального времени (проведение диагностики непосредственно во время опыта) и в локальном режиме (обработка ранее полученных данных).

Полная блок-схема комплекса показана на рис. 2.

В будущем планируется:

- Довести до стабильного рабочего состояния три интерфейса.
- Улучшить возможность вычисления на внешнем оборудовании (на кластере).

- Разработать систему автоматического определения погрешности измерения и геометрических параметров факела пламени.
- Собрать специализированный дистрибутив, который будет содержать в себе платформу TERM и все необходимые инструменты для её модернизации и разработки.
- Разработать сайт платформы TERM:
  - рассылку;
  - документацию;
  - git-репозиторий.

Проект открыт для всех. Будем рады новым идеям и предложениям. Связь через e-mail `mailto:m31@root-ua.com` или джаббер `m31@jabber.od.ua`.

Денис Силаков, Владимир Рубанов Москва, ИСП РАН

Проект: LSB SDK
`http://ispras.linux-foundation.org/LSB_DB_Navigator`

LSB SDK — инструментарий разработки переносимых Linux приложений

Аннотация

В докладе представлен один из элементов инфраструктуры, разрабатываемой в ИСП РАН совместно с Linux Foundation для независимых разработчиков Linux-приложений — LSB Software Development Kit (SDK) — инструментарий, основанный на использовании для компиляции и компоновки приложений специальных заголовочных файлов и библиотек, гарантирующих отсутствие среди зависимостей получаемой программы непереносимых интерфейсов. Также SDK позволяет создавать приложения для «legacy» дистрибутивов, имеющих принципиальные отличия от системы, в которой производится сборка.

Контроль внешних зависимостей приложения

Внешние зависимости программ, собираемых из исходного кода, не всегда полностью контролируются разработчиками — ряд зависимостей может быть обусловлен переменными окружения, версиями

компилятора и библиотек и другими внешними факторами. При этом такие зависимости могут ограничивать переносимость приложения, сужая круг систем, где оно способно функционировать.

Некоторые характеристики среды сборки изменить достаточно сложно — например, при динамической компоновке с библиотекой, использующей версионирование символов, для результирующего файла в качестве зависимостей проставляются символы с версиями «по умолчанию». Однако эти зависимости будут препятствовать запуску приложения на более старых системах, в которых таких версий ещё не было. Обойти эту проблему можно, лишь используя более старую библиотеку при компоновке; но в большинстве случаев в системе присутствует только одна версия библиотеки, а ручное добавление более старой версии может быть нетривиально ввиду необходимости удовлетворить зависимости старой библиотеки.

Поэтому даже если разработчики осведомлены о наличии внешних факторов, привносящих нежелательные зависимости в их приложения, избавление от таких факторов и зависимостей может потребовать значительных усилий.

Одним из возможных путей решения проблемы контроля зависимостей и повышения переносимости приложений является использование инструментария LSB SDK, позволяющего с использованием одной и той же среды сборки получать программы с различными характеристиками, предназначенные для запуска в разных поколениях дистрибутивов Linux.

Состав LSB SDK

LSB SDK включает три основных компонента:

- заголовочные файлы;
- библиотеки-заглушки;
- обёртки для компиляторов C и C++ (lsbcc и lsbc++ соответственно).

SDK позволяет собирать приложения, отвечающие требованиям любой версии стандарта Linux Standard Base (см. `http://ldn.linuxfoundation.org/lsb`), начиная с 3.0. Заголовочные файлы LSB SDK содержат декларации только тех функций, которые включены в одну из версий стандарта LSB, а значит только таких, использование которых является переносимым на большинстве популярных дистри-

бутивов. В файлах определяются все необходимые для использования функций типы данных и константы, а также макросы, которые не приводят к появлению зависимостей от нестандартизованных функций.

Библиотеки-заглушки экспортируют только бинарные символы, включённые в стандарт. Для каждой версии стандарта SDK содержит отдельный набор библиотек-заглушек.

В процессе своей работы lsbcc и lsbc++ вызывают системный компилятор, но при компиляции и компоновке используются библиотеки и заголовочные файлы из SDK.

На основе опции `--lsb-target-version` (задающей целевую версию LSB) выставляется значение переменной среды, влияющей на доступность деклараций внутри заголовочных файлов, а также выбирается набор библиотек-заглушек, с которыми необходимо производить компоновку.

В зависимости от целевой версии LSB, выставляются различные опции компилятора и компоновщика — запрет или разрешение использования «stack protection» в функциях glibc, выставление (при необходимости) `--hash-style` в `sysv` и т. п.

Каждая версия LSB соответствует определённому поколению основных дистрибутивов Linux (LSB 3.0 — RHEL 4.2, SLES 10.0, Mandriva 2006, LSB 3.1 — RHEL 5, Mandriva 2007.0, SLES 10.1, и т. п.). Поэтому разработчики могут выбирать версию LSB, ориентируясь на целевые дистрибутивы, в которых должно функционировать их приложение.

Использование LSB SDK

Использование LSB SDK прозрачно для разработчиков — в большинстве случаев достаточно вместо системного компилятора просто использовать lsbcc (lsbc++ для программ на C++), например, посредством присвоения переменной CC значения «lsbcc» (а переменной CXX — значения «lsbc++»). Для проектов, использующих pkg-config, предоставляются .pc- файлы, при использовании которых pkg-config возвращает информацию, соответствующую LSB SDK, а не системным компонентам. Также поддерживается совместная работа с libtool.

Использование элементов, не входящих в LSB

LSB SDK позволяет осуществлять компоновку с библиотеками, не входящими в LSB, с помощью опции `--lsb-shared-libs`. Опции `--lsb=libpath` и `--lsb-includepath` позволяют указать пути к библиотекам и заголовочным файлам, которые следует использовать при компиляции и компоновке вместо тех, что входят в состав LSB SDK. В будущем планируется реализация режима «relaxed mode», позволяющего использовать не-LSB символы из библиотек, включённых в стандарт.

По умолчанию программы, собранные с помощью LSB SDK, используют в качестве загрузчика ld-lsb. Это поведение может быть изменено с помощью опций `--lsb-use-default-linker` и `--lsb-besteffort`, при использовании которых в качестве требуемого загрузчика указывается ld-linux. Для приложений, собранных с опцией `--lsb-besteffort`, реально используемый загрузчик выбирается при старте приложения — если возможно, то используется ld-lsb, в противном случае — ld-linux.

Вместо вышеперечисленных опций возможно использование соответствующих переменных среды, информацию о которых можно найти в справке lsbcc.

LSB Eclipse Plugin

Инструментарий LSB SDK может быть интегрирован в среду разработки Eclipse с помощью соответствующего плагина, предоставляемого Linux Foundation (`http://ispras.linuxfoundation.org/index.php/About_LSB_Eclipse_Plugin`). Плагин добавляет в Eclipse следующие виды проектов:

- LSB Executable program (C/C++);
- LSB Shared Library (C/C++);
- LSB Static Library (C/C++).

Проекты являются аналогами соответствующих стандартных проектов Eclipse, однако сборка осуществляется с помощью lsbcc и lsbc++. Поддерживается преобразование проектов из «обычных» в LSB и обратно.

Плагин позволяет осуществлять настройку всех опций lsbcc и lsbc++ непосредственно из Eclipse. В частности, могут быть заданы версия LSB, указаны не входящие в LSB библиотеки, с которыми

следует производить компоновку, заданы дополнительные опции для системного компилятора.

Помимо интеграции инструментария LSB SDK, плагин позволяет с помощью Linux Application Checker анализировать совместимость собранного приложения с различными дистрибутивами Linux непосредственно во встроенном веб-браузере Eclipse.

Для функций, используемых в приложении, плагин позволяет отображать в браузере Eclipse информацию, получаемую от LSB Navigator (`http://linuxfoundation.org/navigator`): сигнатуры, ссылки на документацию, данные о присутствии в дистрибутивах и использовании в приложениях и другие сведения.

Заключение

Изначально основной целью LSB SDK была помощь разработчикам в создании приложений, совместимых со стандартом LSB. Однако в настоящее время функциональность, предоставляемая инструментарием, существенно шире. Гибкость и разнообразие режимов работы SDK позволяют разработчикам использовать его для повышения переносимости своих приложений, не ограничиваясь при этом рамками LSB.

Александр Терентьев Нижний Новгород, ООО «Информационный вычислительный центр»

Проект: «Научный калькулятор» www.hopcomp.net

Удалённые научные вычисления — проект «Научный калькулятор»

Аннотация

Доклад посвящён проекту создания комплекса программного обеспечения для выполнения вычислительно сложных научно-технических расчётов с использованием графических процессоров

Целью и основным содержанием проекта является разработка средств, позволяющих упростить использование суперкомпьютерных технологий в научных и инженерных расчётах.

Методология современных наук включает, в том числе, и широкое применение вычислительных методов как для обработки экспериментальных данных, так и для математического моделирования систем и процессов. Объёмы обрабатываемой информации и сложность исследуемых объектов постоянно растут. Более того, многие задачи вообще имеют смысл только в случае возможности обработки данных в реальном времени. То есть в условиях реального отсутствия доступа к суперкомпьютерам приходится сознательно ограничивать полноту и сложность моделей и, соответственно, качество решений.

В то же время существует общедоступная вычислительная архитектура, обеспечивающая при минимальных затратах терафлопсную производительность на настольных системах, а именно графические процессоры. Однако возможность их практического использования для решения прикладных (научных и инженерных) задач сильно ограничено отсутствием удобных программных инструментов и высокой сложностью разработки эффективного программного обеспечения.

Для разрешения существующей неудовлетворительной ситуации мы предложили подход, включающий:

1. использование общедоступной высокопроизводительной вычислительной платформы;
2. промежуточный программный слой, который должен скрыть от пользователя-непрограммиста сложности вычислительной архитектуры;
3. средства удалённого и распределённого доступа.

Задача проекта — радикально упростить вычисления, оставив при этом учёному полную свободу в разработке методов и интерпретации результатов.

На первом этапе проекта мы сосредоточили основные усилия на разработке структуры и внутренних интерфейсов программного обеспечения, а также средств синхронизации, обеспечивающих высокий уровень модульности и независимости вычислительных компонентов и позволяющих скрыть от пользователя специфику параллельной архитектуры.

В качестве аппаратного обеспечения используются высокопроизводительные графические процессоры семейства GeForce.

В рамках проекта разрабатываются:

- Унифицированный программный интерфейс;
- Средства синхронизации данных и результатов вычислений;
- Графический интерфейс пользователя;
- ПО вычислительных функций (на сегодняшний день реализованы решение дифференциально-разностных уравнений, фрактальная динамика, нейросетевые алгоритмы).

Средства интерактивного пользовательского интерфейса обеспечивают быструю сборку цепочки вычислительных функций, их настройку (задание параметров) и непосредственное выполнение вычислений. Программный интерфейс обеспечивает также вызов вычислительных функций из оригинальных прикладных программ.

В ближайшее время будет опубликовано описание программных интерфейсов, что позволит привлечь к разработке вычислительных и других функций максимальное количество специалистов: как программистов, так и учёных-прикладников.

Михаил Якшин Москва, Inquisitor team

Проект: Einarc `http://www.inquisitor.ru/doc/einarc/`

Einarc — автоматизация управления аппаратными RAID-контроллерами

Аннотация

До сих пор любой, кто хотел автоматизировать управление своими аппаратными RAID-массивами, как правило, вынужден был скачивать для этого специальную проприетарную утилиту, изучать её и затем использовать. Einarc — решение для унификации всех существующих (в том числе проприетарных) парадигм управления устройствами хранения информации. Einarc работает в качестве транслятора и даёт возможность пользователю оперировать простыми и хорошо определёнными терминами — такими, как «физический диск», «логический диск», «адаптер» и т. п. Запросы в терминах Einarc на лету преобразуются в команды проприетарных CLI, а ответы затем преобразуются обратно. Система продолжает пользоваться решениями производителей контроллеров (что обеспечивает максимальную совместимость),

но пользователь (или программист) не общается с ними напрямую, оставаясь в рамках единого, хорошо документированного интерфейса.

История возникновения и использования RAID-массивов достаточно обширна и хорошо освещена в литературе. Концепция RAID (Redundant Array of Independent Discs) — применение массива из нескольких относительно ненадёжных, небольших и медленных жёстких дисков для обеспечение массива высокой надёжности, ёмкости и скорости — появилась в конце 1980-х годов и в настоящий момент имеется в арсенале любого достаточно квалифицированного системного администратора.

Рынок RAID делится на две большие части: аппаратные и программные решения. Аппаратные решения представляют собой специализированные контроллеры, к которым можно подключать жёсткие диски, а из прикладных программ обращаться к неким «виртуальным» жёстким дискам, которые будут соответствовать сконфигурированным массивам. Программные решения, доступные в большинстве популярных ОС, организуют такие «виртуальные» диски.

Общеизвестны и достоинства, и недостатки обоих подходов [1].

Единственный, но существенный недостаток программных реализаций в том, что они работают достаточно медленно в режимах RAID 5/6 и подобных им, где требуется обсчёт большого количества операций XOR на записываемом/считываемом потоке данных. В связи с этим практическое использование программных RAID обычно ограничивается массивами типа RAID 0, 1 и их комбинациями (RAID 0+1, RAID 10).

«Настоящие» аппаратные решения достаточно дороги — в первую очередь из-за того, что обязаны включать в себя:

- специализированный сопроцессор, который может быстро считать XOR'ы, необходимые для реализации RAID 5 и его аналогов;
- память для использования в качестве кэша,
- аккумуляторную батарею, которая будет использоваться для поддержания питания памяти при использовании режимов write back [2],
- основной процессор и инфраструктру для управления всем этим комплексом — для снятия нагрузки с процессора и системных шин.

Стоимость самых дешёвых аппаратных RAID-контроллеров на момент написания статьи (порядка 300–400 USD [3]) сопоставима со сто-

имостью 4–5 ТВ дискового пространства. Где-то здесь проходит экономическая граница целесообразности: при создании систем из N (где N начинается от 6–7) дисков разумнее потратить деньги на аппаратный RAID-контроллер и получить N-1 диск полезного пространства (использовав RAID 5), а не N/2 диск полезного пространства (использовав программный RAID 10). Нужно заметить, что кроме таких факторов, как стоимость контроллера и дисков, стоит учитывать стоимость и размеры корпуса с корзинами/backplane под большое число дисков, блока питания, который сможет выдавать такие мощности, а также систем охлаждения.

Второй недостаток аппаратных RAID-контроллеров — ввиду определённой специфики индустрии и жёсткой конкуренции в этом сегменте, все существующие на данный момент контроллеры являются проприетарными и несовместимыми друг с другом. Это относится и к форматам служебной информации, записываемой в память и на диски, и — самое главное — к интерфейсам управления массивами, предоставляемым системным администраторам.

Традиционная картина, которую можно наблюдать на предприятиях со средним и большим парком серверов, а также зачастую у хостинг-провайдеров, такова. Устанавливается (абсолютно логично и экономически мотивированно) первый 2U-сервер с 6 дисками в RAID 5 с аппаратным контроллером в качестве большого файл-сервера, почтового сервера, сервера СУБД или чего-то подобного. Через некоторое время (например через полгода) требуется расширить дисковое пространство, но в существующий 2U-сервер нельзя вставить больше дисков — приобретается второй 2U-сервер с 6 дисками, но зачастую с другим контроллером (что может случаться по массе причин — начиная от недосмотра/непонимания серьёзности проблемы и заканчивая некими организационно-экономическими предпосылками). При достаточно активном росте скоро появляются ещё несколько серверов с абсолютно разными RAID-контроллерами (а, возможно, ещё и несколько серверов с программным RAID).

Конфигурирование и обслуживание всех этих аппаратных RAID-контроллеров превращается в ад для системного администратора: приходится изучать массу подходов, интерфейсов и утилит управления и пытаться свести их в единую систему для того, чтобы подключить эти массивы к мониторингу. Хотя базовые драйверы для большинства RAID-контроллеров присутствуют в официальной поставке ядра Linux, управление и мониторинг массивов невозможны

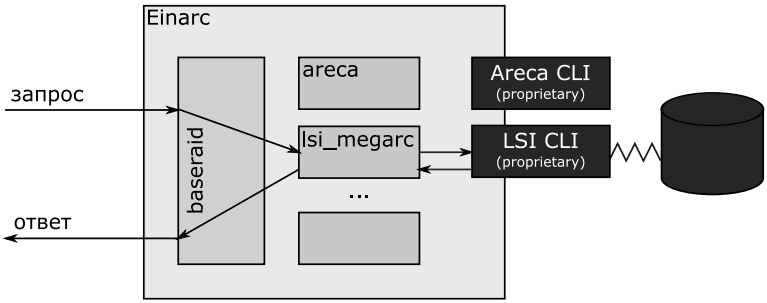

Рис. 1: Общая архитектура Einarc

без применения соответствующих проприетарных утилит от производителей массивов. Не существует даже единой концепции представления объектов, с которыми оперируют такие утилиты: например, Areca использует трёхзвенную иерархию (физические диски → raidset'ы → volumeset'ы), а LSI — двухзвенную (физические диски → логические диски).

Кардинальный способ решения такой задачи — изначально ориентироваться на оборудование только от одного производителя — хоть и несёт в себе массу преимуществ (например возможность пользоваться полноценно всей инфраструктурой менеджмента серверов, предоставляемых этим вендором), к сожалению, не всегда применим, особенно если не запланирован с самого начала.

Комплекс Einarc (Einarc Is Not A RAID CLI) призван решать задачу объединения интерфейсов управления и мониторинга дисковых массивов (в том числе аппаратных и програмных RAID-массивов). При использовании Einarc системный администратор освобождается от отдельного конфигурирования массивов и их мониторинга на каждом сервере, а также от изучения десятка существующих проприетарных утилит и концепций работы с ними. Все команды всех поддерживаемых устройств сведены в единый документированный интерфейс. Например, команда, выводящая список логических дисков (т. е. тех «виртуальных» жёстких дисков, которые обычно видны системе как `/dev/sda` и т. д.) с их состояниями, будет иметь вид `logical list` для всех поддерживаемых контроллеров.

Достигается это за счёт нескольких ключевых понятий Einarc:

Объектная модель Разработана общая объектная модель, которая является расширенным пересечением объектных моделей всех поддерживаемых RAID-контроллеров и покрывает практически все потребности системного администрирования: создание, настройку, мониторинг и удаление массивов.

Транслятор запросов и ответов Einarc образует «оболочку» вокруг проприетарных утилит RAID-контроллеров, тем самым избавляя пользователя от необходимости взаимодействовать с ними напрямую. Пользователь формирует команду в рамках общей объектной модели, затем запрос транслируется на «язык» утилиты RAID-контроллера, выполняется, а ответ транслируется обратно на «языке» общей объектной модели.

Модули поддержки различных видов RAID Einarc не монолитен: поддержка каждой из проприетарных моделей вынесена в отдельный модуль (например, все контроллеры Areca поддерживаются в `areca`, все SATA/SAS-контроллеры LSI — в `lsi_megacli` и т. д.) Написание новых модулей максимально упрощено: необходимо реализовать методы в виде запросов к очередному проприетарному CLI, соответствующих по смыслу объектной модели. На добавление read-only поддержки нового контроллера уходит порядка 1–2 часов, на добавление полноценной поддержки — несколько дней.

Автоматическая инсталляция Для работы Einarc требуются проприетарные утилиты от производителей RAID-контроллеров. К сожалению, лицензия не позволяет перераспространять их вместе с Einarc, но, тем не менее, в пакете приложена масса усилий для облегчения процедуры их инсталляции. И при сборке из исходных текстов, и при использовании бинарного пакета из дистрибутива Einarc приводится в рабочее состояние одной командой типа `einarc-install --modules=auto`, которая автоматически распознаёт установленные RAID-контроллеры, скачивает, распаковывает и устанавливает для них проприетарные утилиты.

Einarc может быть полезен не только в рамках системного администрирования. Как показала практика, Einarc оказался полезным и для систем автоматизированного тестирования (на примере системы Inquisitor [4]), и для систем реализации Network Attached Storage

(NAS) — на примере существующих патчей для проекта OpenFiler и собственной реализации NAS.

Литература

[1] *Martin, B.* Benchmarking hardware RAID vs. Linux kernel software RAID. http://linux.com/feature/140734

[2] Словарь SNIA, статья «Write back cache»: http://www.snia.org/education/dictionary/w/

[3] Сайт pricewatch.com, запрос «hardware raid 5 sata 4 port»: http://www.pricewatch.com/search?q=hardware+raid+5+sata+4+port&cn=Controller+Cards

[4] *Якшин М.М* Inquisitor — свободная платформа тестирования и мониторинга программного обеспечения // На Протве. 5-я конференция разработчиков свободных программ. Обнинск, июль, 21–23. 2008. Тез. докл. — М., Институт Логики: 2008. — С. 17–20

Фёдор Зуев, Институт Земной Коры СО РАН

Копирайт и договор в публичных лицензиях

Якобсен против Катцера

Год назад в США произошло событие, чрезвычайно важное для свободных лицензий. 13 августа 2008 года Апелляционный суд федерального округа США постановил, что действия Мэтью Катцера, удалившего из файлов «описаний декодеров» проекта JMRI ссылки на авторов, проект и лицензию и распространявшего их в составе своей программы Decoder Commander, являются нарушением авторского права. Отменив решение судьи первой инстанции, полагавшего, что эти действия являются всего лишь нарушением договора — Artistic License, под которой распространялся дистрибутив JMRI.

В течение многих лет пропагандисты свободного ПО повторяли, что публичные лицензии надёжны для авторов, поскольку при возникновении проблем происходит откат в исходное состояние «без лицензии» и автор получает в свои руки мощное репрессивное оружие

копирайта. Оказалось, что опасность пришла вовсе не оттуда, откуда её ожидали — не от «непризнания свободных лицензий» а напротив, от слишком расширительного толкования их текста. И хотя это столкновение закончилось победой принципов свободного ПО, в целом вопрос далеко не исчерпан. Чтобы понять, в чём он состоит, надо разобраться в различиях между копирайтом и договором и в границах между ними.

Обязанности из закона и договора

Авторское право — это установленная государством монополия, исключительное право. Договор же — это инструмент, посредством которого граждане сами создают себе права и обязанности. Люди часто не осознают, что обязанности, возникающие из этих источников, различаются не только происхождением, но и свойствами:

Обязательства, основанные на авторском праве	*Обязательства, основанные на договоре*
Ограничены установленным законом перечнем действий, составляющих авторское право	Могут касаться — за вычетом нескольких установленных законом исключений — практически любых действий, которых стороны договора только способны четко сформулировать
Возникают автоматически по факту создания произведения	Заключение договора — формальная процедура, требующая активного сознательного участия всех заинтересованных лиц
Обязательны для всех граждан государства, а посредством международных конвенций — и для большинства иностранцев	Обязывают только подписавших договор лиц. Лица, договор не подписывавшие, его требованиями не связаны, даже если знают о его существовании и содержании

Регулируются законами страны, где обязательства осуществляется. Практически это обычно означает место жительства должника. Никто не может изменить применимую юрисдикцию	Установление применимого законодательства может быть довольно сложной задачей. Стороны могут оговаривать применимое право непосредственно в договоре и такие оговорки могут быть обязательны для суда
За нарушение авторских прав в большинстве стран предусмотрен обширный набор санкций: как денежных, так и личных. Многие из них наказывают за сам факт нарушения, независимо от его последствий	В общем случае нарушитель договора обязан лишь возместить убытки своего контрагента
Выплата штрафа или исполнение иного наказания не освобождает от обязанностей соблюдать авторское право	Нарушение договора ведёт к его расторжению, если только стороны не договорятся об ином

Мы видим, что лицензиар, опирающийся на договорные обязательства, оказывается в менее прочном положении, чем тот, за спиной у кого стоит машина копирайта. Именно в такую ситуацию, собственно, сталкивало Якобсена решение первой инстанции: сначала утверждалось, что отношения Якобсена с Катцером имеют чисто договорной характер, а затем «внезапно обнаруживалось», что Якобсен не в состоянии доказать конкретную сумму убытков.

Где кончается договор и начинается копирайт?

Если между автором и пользователем нет никакого договора, то вопросов, разумеется, не возникает — единственные отношения, которые могут быть между ними, — это отношения, основанные на копирайте. Сложнее, если между ними существует договор об использовании произведения и одновременно у них возникает спор о нарушении прав на это же произведение. А это именно та ситуация, в которой находятся авторы и пользователи свободных программ.

Вопрос может показаться простым, но тем не менее однозначного ответа на него нет, мнения судов различных стран разноречивы и отрывочны.

У нас ещё с советских времен в учебниках в качестве примера приводилась такая ситуация: если издательство выпустило произведение большим тиражом, чем указано в договоре, то это нарушение авторских прав. А если не выплатило в срок гонорар — то это нарушение договора. Однако в 1999 году ВС РФ принял довольно странное решение (№194пв-98пр от 31 марта 1999 г), по которому издательство, вовсе не использующее произведение (и как раз из-за неиспользования) было признано виновным в нарушении именно авторского права.

Американское законодательство в целом не признаёт личного права автора на имя. Но если требование указания имени имеется в лицензионном договоре, его отсутствие рассматривается как нарушение копирайта, а не нарушение договора.

Копирайт и договор в мире проприетарного ПО

В особенности тень на плетень в этом вопросе наводят проприетарные софтоиздатели, которые вот уже двадцать с лишним лет посредством пресловутых EULA явочным порядком «продавливают» себе универсальное «софтверное право», сочетающее наиболее выгодные элементы различных исключительных прав и договорных обязательств. Таким образом, чтобы оно распространялось на всё, что вздумается издателю, было общеобязательным и влекло суровые санкции за нарушение. И нельзя сказать, чтобы эти усилия были совсем уж бесплодными, напротив. Во многих странах (прежде всего в США) накопился ворох весьма странных судебных прецедентов. В принятую в 2006 году «IV часть Гражданского Кодекса» включён ряд странных клауз, которые, в принципе, можно толковать в рамках традиционного исключительного права, но в этом случае формулировки получаются очень неестественными.

Мы не можем полагаться на эти достижения конкурентов при оценке и конструировании публичных лицензий по ряду причин. Во-первых, по большей части они вовсе не стремятся к проведению чёткой границы между законом и договором, а, напротив, направлены на размытие этой границы. Во-вторых, по большей части эти решения основываются непосредственно на экономическом или политическом давлении, юридические же их основания весьма шатки, и поэтому они легко могут быть аннулированы изменением политического курса.

Игорь Воронин Шатура, ИПЛИТ РАН

Проект: Беспроводные распределённые сенсорные сети
http://wiki.laser.ru/index.php/
Беспроводные_распределённые_сенсорные_сети

Исследование вариантов сбора данных в Распределённых Сенсорных Сетях на основе СПО

Аннотация

В предлагаемом докладе пойдёт речь о перспективном направлении современной телекоммуникационной отрасли — беспроводных распределенных сенсорных сетях (РСС). Будет рассказано об основных принципах работы, преимуществах и достоинствах в их использовании. Особый акцент будет сделан на программное обеспечение — осуществляющее сбор данных в сенсорных сетях на основе СПО. Предложены варианты проведения лабораторных работ для уроков физики в старших классах общеобразовательной школы на основе РСС. Будут озвучениы аспекты по развитию технического творчества учеников, предложены конкретные шаги в этом направлении.