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

Восьмая конференция разработчиков свободных программ

Обнинск, 25–26 июля 2011 года

Тезисы докладов

Москва,
Институт Логики,
2011

УДК 004.91
ББК 32.97

Восьмая конференция разработчиков свободных программ: Тезисы докладов / Обнинск 25–26 июля 2011 года.
М.: Альт Линукс, 2011. — XX с. : ил.

В книге собраны тезисы докладов, одобренных Программным комитетом Восьмой конференции разработчиков свободных программ.

ISBN 978-5-905167-07-2

Программа конференции

25 июля

11.50–12.20 Регистрация, кофе, чай

Дневное заседание
12.20–14.00

А. Е. Новодворский Вступительное слово. Открытие. Информация оргкомитета

Ilkka Lehtinen

Open Source inside public sector in Finland and EU 6

Евгений Синельников

Публичный сборочный сервер Etersoft 6

Михаил Пожидаев

Дистрибутив со вспомогательными технологиями ALT Linux Homeros Friend: первый опыт 9

14.00–15.00 Обед

Вечернее заседание
15.00–19.00

Алексей Гладков

Новый тип расширений Mozilla (Jetpack) 12

Константин Лепихов

Organic software. Как это работает 13

Андрей Михеев

Реализация бинарных отношений в свободной системе управления бизнес-процессами и административными регламентами RunaWFE для упрощения инициализации ролей 14

26 июля

Дневное заседание
10.00–14.00

14.00–15.00 Обед

Вечернее заседание
15.00–17.00

Михаил Шигорин

17.00–18.30 Круглый стол «Перспективы разработки СПО в России и НПП» Ведущий А.Е. Новодворский

Вне программы

Григорий Злобин

Ilkka Lehtinen
Finland/Tampere, The Finnish Centre for Open Source Solutions
Проект: COSS `http://www.coss.fi/`

Open Source inside public sector in Finland and EU

Presentation will give an short overview of Open Source market situation in Finland and EU, especially in cases where user organization in public sector office. (governmental, municipal, etc). How Open Source solutions are used today in Finnish business life, how providers and customers are acting with each other, what are motivation drivers to use open source solutions? Presentation will also introduce different stakeholders of Open Source market. It has been difficult for public sector organizations to interact or contribute to Open Source communities. What are the practical solutions for this problem? Osor.eu as a phenomena - internet is full of different repositories for software. Why did European Commission build yet another one? Or is it more than yet another one?

Евгений Синельников
Саратов, ООО «Этерсофт»
Проект: Korinf http://freesource.info/wiki/korinf

Публичный сборочный сервер Etersoft

Аннотация

В докладе рассмотрены возможности сборочной песочницы для публичных проектов, предоставляемой компанией Etersoft, для сборки пакетов под произвольные целевые операционные системы на основе единого src.rpm. Представлено описание проект Korinf и сборочных технологий ALT Linux на основе которых выполнен сервер. Приведены расширения, внесённые в проекты girar и girar-builder с целью их использования в частных, рабочих окружениях. Представлены возможности и ограничения по использованию сервера и администрированию проектов.

Применение сложных решений в простых задачах в сфере информационных технологий встречается постоянно. В некоторых задачах это оправдано простотой использования, в других — возможностями быстрого наращивания функциональных возможностей за счёт использования модульной архитектуры, в третьих — широкими возможностями масштабирования. Сборочная инфраструктура для Linux-решений тоже, в этом плане, является сложным решением.

Компания Etersoft с 2005 года применяет систему сборки пакетов Korinf для внутренних задач по подготовке к продажам продукта WINE@Etersoft, а также применяет это решение для сборки и отгрузки всех продаваемых продуктов по заявкам в автоматическом режиме. В результате применения данной системы осуществляется поддержка широкого числа систем:

WINE@Etersoft 1.0 SQL		**SELTA@Etersoft 1.1**	
Полная поддержка	Базовая поддержка	Полная поддержка	Базовая поддержка
ALTLinux с 4.0 по p6	ALTLinux Sisyphus	ALTLinux с 4.0 по p6 и 64-bit	ALTLinux Sisyphus
ASPLinux 14	ASPLinux 12	ASPLinux 14	ALTLinux 4.0 и 4.1 64bit
Debian 5.0, 6.0 и64-bit	ASPLinuxServer V	Debian 5.0, 6.0 и64-bit	ASPLinux 12
Fedora с 11 по 15	Debian 4.0	Fedora с 11 по 15 и 64-bit	ASPLinuxServer V
LINUX@Etersoft 4.1	ArchLinux 2010.05	Mandriva с 2009.1 по 2010.2 и 64-bit	Debian 4.0
Mandriva с 2009.1 по 2010.2	Gentoo 2009	CentOS 5 и 6	ArchLinux 2010.05
CentOS 5 и 6	InfraLinux 8.10 и 9.04	Scientific 6	Gentoo 2009
Scientific 6	MOPSLinux 6.0	SUSE с 11 по 11.3 и 64-bit	InfraLinux 8.10 и 9.04
RHEL 5 и 6	Mandriva 2008.1 и 2009.0	Ubuntu с 6.06 по 11.04 и 64-bit	LinuxXP 2008
SUSE 11.3	PCLinux 2009.2	Windows	Runtu 1.1
Ubuntu с 8.04 по 11.04 и 64-bit	Runtu 1.1		RHEL 5 и 6
	StartCom 5		Slackware 12
	SLED 11 и SLES 11		StartCom 5
	SUSE 11.1 и 11.2		Ubuntu 8.10 и 64bit
	Slackware 12		
	Ubuntu (6.06 и 8.10)		

Публичный сборочный сервер (Public Build Farm) — это решение, которое позволяет осуществить сборку из git-репозитория в Korinf под заданный набор систем. Консольный интерфейс, предоставляемый пользователю для управления и сборки пакетов, представляет собой модифицированный вариант оболочки girar, в которую включён специальный вариант сборочной задачи, что позволяет использовать стандартные механизмы управления задачами, используемыми в git.altlinux.org [1] и git.etersoft.ru [2].

Текущее состояние публичного сборочного сервера требует отладки и доработки, а также выработки типовых вариантов использования и документирования. Подробное описание расширенного интерфейса управления можно найти на странице документации проекта [3]. На текущий момент интерфейсы управления ещё не утверждены и могут быть со временем изменены. Кроме того, к публичному сборочному серверу планируется реализовать web-интерфейс для управления ssh-ключами, уведомлениями и настройками сборочных сред и систем.

Стоит учесть ряд важных принципов, которые определяют возможности ограничения публичного сборочного сервера Etersoft. Основные принципы применения включают в себя следующее:

- сервер не предназначен для сборки системообразующих пакетов (не стоит применять Коринф для сборки glibc или rpm для разных систем);
- сборочные среды для отдельных систем ограничены и требуют ручной настройки по запросу для расширения;
- сервер предназначен для сборки под разные системы только одного продукта и не приспособлен для сборки наборов зависимых приложений и библиотек.

Применению публичного сборочного сервера включают в себя возможность по решению следующих задач:

- Создание дистрибутиво-специфичных репозиториев бинарных пакетов;
- Тестовая пересборка пакета«под все системы» (полезно для тестирования разработчиком).

Таким образом, проект, в первую очередь предназначен для конечных разработчиков, желающих публиковать и распространять свои решения. Базовой средой разработчика, для которой предназначен

публичный сборочный сервер, является ALT Linux Sisyphus, а также, возможно, Национальная Программная Платформа.

Рассмотренный в докладе публичный сборочный сервер представлен с целью предоставления возможностей по распространению свободных решений, а также для популяризации и ознакомления с услугами предоставляемыми компанией Etersoft.

Литература

[1] ALT Linux Team, git.alt — сервер совместной разработки ALT Linux Team, `http://www.altlinux.org/Git.alt`

[2] Etersoft, Uses Git Etersoft — руководство пользователя Git Etersoft, `http://wiki.etersoft.ru/UsesGitEtersoft`

[3] Etersoft, Public Build Farm — публичный сборочный сервер Etersoft, `http://wiki.etersoft.ru/admin/pbf`

Михаил Пожидаев
Томск, Томский государственный университет
Проект: ALT Linux Homeros `http://homeros.altlinux.org`

Дистрибутив со вспомогательными технологиями ALT Linux Homeros Friend: первый опыт

Аннотация

В докладе освещаются основные концепции и возможности дистрибутива ALT Linux Homeros Friend, который за счёт достаточно скромных требований к системным ресурсам компьютера может рассматриваться как хорошая операционная система для запуска на мобильных устройствах, предназначенная для использования людьми с ограничениями по зрению.

ALT Linux Homeros Friend — это операционная система, предназначенная для использования людьми с ограничениями по зрению. Помимо ALT Linux Homeros Friend в будущем планируется представить ALT Linux Homeros Desktop, основой которого послужит оконная среда GNOME.

Дистрибутив ALT Linux Homeros Friend создан на базе текстового редактора GNU Emacs, гибкие возможности которого позволяют создать очень удобное окружение для выполнения большого числа повседневных задач. GNU Emacs способен представить множество рабочих объектов пользователя в текстовом виде, что хорошо подходит для построения системы, ориентированной на использование при помощи речевого интерфейса. Кроме этого, среда GNU Emacs обладает крайне скромными требованиями к системным ресурсам компьютера, что позволяет использовать операционные системы на её основе на таких мобильных устройствах, как ноутбуки и нетбуки.

К текущему моменту удалось найти решение достаточно большого числа задач, позволяющих приблизить дистрибутив ALT Linux Homeros Friend к уровню законченной операционной системы, подходящей для использования в повседневной работе. В дистрибутиве работа начинается с открытия главного меню, производимого путём нажатия соответствующей клавиши в нижней части многих клавиатур. В его верхней части отображаются текущее время, дата и уровень заряда аккумулятора в случае использования мобильного компьютера. Ниже следует раздел с пунктами, позволяющими открыть директорию с документами, календарь, адресную книгу, списки книг для чтения (из звуковых и текстовых файлов), менеджер сменных носителей, почтовый клиент, веб-браузер и пр.

Вывод речевой информации производится при помощи речевого сервера VoiceMan, разработанного непосредственно в рамках проекта ALT Linux Homeros. Речевой сервер VoiceMan обладает удобными возможностями конфигурирования, включая функцию «горячего» переключения синтезаторов без потери клиентских соединений. Основным синтезатором для обработки русской речи в настоящий момент является RHVoice.

Для воспроизведения музыкальных файлов и «говорящих книг» в рамках проекта ALT Linux Homeros ведётся создание специальной утилиты Musitorius, которая является сервисом, выполняющим управление медиапроигрывателем MPlayer. Использование Musitorius позволяет посылать команды проигрывателю децентрализованно, что облегчает применение мультимедийных клавиатур. Утилита снабжена механизмом протоколирования проигранных файлов, позволяющим вернуться к прослушиванию прерванного фрагмента. Такая возможность является ключевой при прослушивании “говорящих книг”.

Несмотря на специализированный характер системы, дистрибутив оснащён некоторыми функциями, выполняющими вывод графической информации на экран. Эти функции включают в себя показ видеофайлов и фотографий, что достаточно часто является удобным в кругу семьи или друзей. Также есть возможность показа в полноэкранном режиме pdf-файлов. Это крайне полезно, если требуется использовать мобильный компьютер для показа презентации на конференции или семинаре.

Наибольшие затруднения в настоящий момент составляют работа с офисными документами и просмотр веб-страниц. Эти функции могут выполнять офисный пакет OpenOffice.org и браузер Mozilla Firefox, но их работа с активированными вспомогательными возможностями доступна только в среде GNOME с экранным чтецом Orca. Такой подход планируется применить в дистрибутиве ALT Linux Homeros Desktop, но в среде на основе GNU Emacs не используется не только настольное окружения GNOME, но даже и X-сервер. Предполагается, что работа проходит сугубо в режиме текстовой консоли, с временным запуском X-сервера по необходимости при воспроизведении видеофрагментов или pdf-файлов.

Таким образом, использовать стандартные подходы невозможно и необходимо исследовать альтернативные решения. В случае работы с офисными документами есть два дополнительных подхода: применение «облачной» службы Google Docs, для которой ведётся подготовка интерфейса внутри GNU Emacs, и применение издательской системы latex, подготовка входной информации для которой выполняется сугубо в текстовой форме. Использование latex целесообразно в случае набора крупных работ, таких как диссертация, книга и пр., в то время как Google Docs должен быть удобен для значительно меньших по объёму материалов.

Основным недостатком веб-браузера w3m, входящего в состав среды GNU Emacs, является невозможность исполнения скриптов Javascript. Это значительно сужает круг открываемых им сайтов, и приемлемых способов устранения этой проблемы пока не известно. Как и в случае работы с офисными документами это ограничение можно смягчить за счёт применения библиотек для взаимодействия с API различных крупных сайтов.

Алексей Гладков
Москва, Mozilla Russia Team
https://developer.mozilla.org/en/Jetpack

Новый тип расширений Mozilla (Jetpack)

Своей популярностью Firefox во многом обязан лёгкому расширению и изменению функционала — расширениям. С ростом популярности продуктов Mozilla стали выявляться как сильные, так и слабые стороны способа реализации расширений. Стало ясно, что желающих писать расширения значительно больше, чем людей владеющих достаточным количеством знаний в технологиях, необходимых для создания расширений для Mozilla.

Чтобы сейчас создать обычное расширение необходимо владеть:

- XUL;
- XPCOM/XPIDL;
- C++;
- Javascript;
- CSS;
- HTML.

Хорошо видно, что преобладают технологии, не знакомые обычным web-разработчикам.

Более того, любое расширение выполняется как часть браузера т.е. имеет возможность обращаться к разным подсистемам движка, читать/писать файлы на файловой системе и выполнять программы. Это порождает много вопросов связанных с безопасностью.

Такая сильная интеграция с браузером приводит к ещё одной проблеме — проблеме совместимости. Многие знают как при выходе очередной версии Firefox многие приложения перестают работать даже если исправить зависимости.

Всё это затрудняет и замедляет написание и поддержку расширений.

Большинству расширений, на самом деле, не нужна такая сильная интеграция с браузером и такие широкие привилегии.

Что тут можно сделать ?

Jetpack — это новый способ создания расширений для Mozilla.

Основными целями Jetpack являются:

- Создать интерфейс, который не зависит от версии движка mozilla;
- Изолировать расширения от реальной системы;
- Предоставить простое API только для языка Javascript;
- Максимально использовать web-технологии.

В итоге был создан высокоуровневый API с использованием CommonJS, который позволяет избежать потери совместимости при смене версии Firefox, а также позволяет повторно использовать один и тот же код другими расширениями.

Расширения исполняются в своей песочнице и не имеют доступа к файловой системе. Они работают только с определёнными частями браузера.

Код расширений пишется с использованием Javascript, HTML и CSS. Это позволяет не изучать специфичных для mozilla технологий и языков. После установки таких расширений не требуется перезагрузка браузера.

Константин Лепихов
Москва, Mozilla Team
Проект: Mozilla `http://mozilla-russia.org`

Organic software. Как это работает

Концепция «органического» программного обеспечения была рождена в недрах Mozilla как способ существовать на рынке закрытых программных решений и вести успешную конкуренцию с ними. Главная идея подхода — программное обеспечение, свободное от патентов и встроенных бинарных объектов, которое с уважением относится к пользователю. Однако кроме пользователя, уважительное отношение задекларировано также к открытым стандартам, дающим пространство для того, чтобы любой индивидуум, компания или организация могли создавать для других веб-контент. Это проявление веры Mozilla в важность предоставления средств для участия в развитии Интернет.

Андрей Михеев
Москва, Консалтинговая группа РУНА
Проект: RunaWFE http://wf.runa.ru/rus

Реализация бинарных отношений в свободной системе управления бизнес-процессами и административными регламентами RunaWFE для упрощения инициализации ролей.

Аннотация

В докладе рассказывается про применение математического понятия «бинарное отношение» к инициализации ролей в системах управления бизнес-процессами и реализацию этого подхода в системе RunaWFE

Система RunaWFE

RunaWFE — открытая, масштабируемая, ориентированной на конечного пользователя система управления бизнес-процессами и административными регламентами. Система платформонезависима (написана на Java), распространяется под LGPL-лицензией.

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

Роли и их инициализация

Исполнителями заданий бизнес-процесса могут быть как сотрудники предприятия, так и информационные системы. Связывание узлов бизнес-процесса с исполнителями заданий производится при помощи ролей. При разработке бизнес-процесса создается роль и ставится в соответствие определенным узлам схемы. Инициализация роли — это назначение на роль конкретного исполнителя. В докладе показано, как концепцию бинарных отношений [1] можно использовать для построения простого механизма инициализации ролей.

Понятие «бинарное отношение»

Бинарное отношение можно рассматривать как расширение понятия функция.

Определение. Бинарным отношением между множествами A и B называется любое подмножество P декартова произведения множества A на множество B. Часто, чтобы обозначить принадлежность упорядоченной пары (a, b) к бинарному отношению P вместо записи $(a, b) \in P$ используют обозначения $P(a, b)$ или aPb. При этом говорят, что a находится в отношении P к b.

Замечание 1. Для множеств A и B, состоящих из конечного числа элементов, любое отношение можно задать, определив набор упорядоченных пар (a, b) для этого отношения.

Замечание 2. Некоторые (но не все) бинарные отношения соответствуют функциям. То есть некоторые бинарные отношения являются функциями. Можно определить функцию как такое бинарное отношение R, в котором каждому значению b отношения aPb соответствует лишь одно единственное значение a (но не наоборот). В этом случае $a = f(b)$, где f — функция, соответствующая бинарному отношению P.

Применение понятия «бинарное отношение» к инициализации ролей

Предлагается кроме традиционных способов инициализации ролей добавить возможность инициализации ролей при помощи бинарных отношений.

Во-первых, это даст возможность инициализировать роль сразу множеством возможных исполнителей заданий. Часто в бизнес-процессе задание направляется не одному исполнителю, а множеству возможных исполнителей задания. Выполняет это задание тот пользователь, который первым возьмет его на исполнение.

Во-вторых, при использовании отношений процедура задания возможных исполнителей задания становится очень простой и ее легко реализовать прямо в графическом интерфейсе.

Отношение над исполнителями заданий предлагается строить при помощи задания набора пар (Исполнитель1, Исполнитель2). При этом не требуется проверять каких-либо ограничений (как, например, для

функции — что она возвращает только одно значение для одного исполнителя).

Использование групп пользователей при задании отношений.

Задавать отношения перечислением всех определяющих его пар пользователей неудобно, так как таких пар может быть очень много. Для уменьшения количества вводимых данных имеет смысл воспользоваться группами пользователей.

Группы пользователей служат для объединения пользователей по какому-либо признаку. Обычно группа «наследует» свойства всех групп, в которые она входит.

Зададим отношение как множество пар (Исполнитель1, Исполнитель2), в которых Исполнитель является пользователем или группой пользователей.

Инициализация роли при помощи отношения производится следующим образом:

1. Из указанной в инициализаторе роли переменной бизнес-процесса берется ее значение-Исполнитель — имя пользователя или группы пользователей. Это значение будет соответствовать правой части отношения.
2. Строится множество значений всех левых частей отношения, соответствующих данному элементу правой части. Делается это так: Для Исполнителя — значения правой части отношения находятся все группы, в которые он входит (хотя бы в одну из их подгрупп). Далее находятся все пары определенные для данного отношения, у которых в правой части стоит Исполнитель или одна из найденных групп. Далее рассматривается множество всех левых частей этих пар.

Если пар нет, то роль не инициализируется. Если множество состоит только из одного пользователя, то роль инициализируется им. В остальных случаях роль инициализируется множеством всех пользователей, попавших в левые части пар или принадлежащих какой-либо из групп попавших в левую часть пар, или какой-либо из их подгрупп.

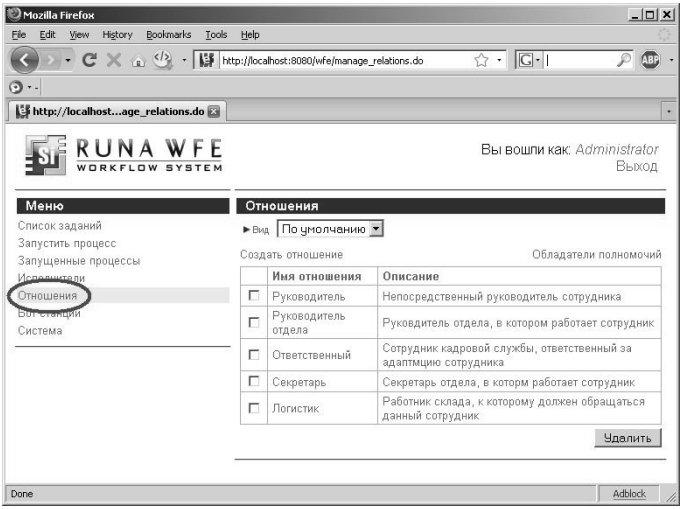

Рис. 1.

Результаты применения.

Концепция отношений реализована в интерфейсе RunaWFE следующим образом

1. В главном меню системы появился еще один пункт меню — Отношения (в английской локализации — Relations см. Рис 1).

В этом пункте можно посмотреть/добавить/удалить отношение, открыть отношение и отредактировать множество составляющих его пар.

2. Для каждого исполнителя в его свойствах добавлены два раздела (см. Рис 2):

- Отношения, в которых он может находиться в левой части
- Отношения, в которых он может находиться в правой части

Каждое отношение можно открыть и отредактировать множество исполнителей в другой части отношения (см. Рис 3)

Работа с отношениями в редакторе бизнес-процессов

В редакторе в бизнес-процессе при редактировании инициализатора роли можно выбрать закладку «задать роль с помощью отноше-

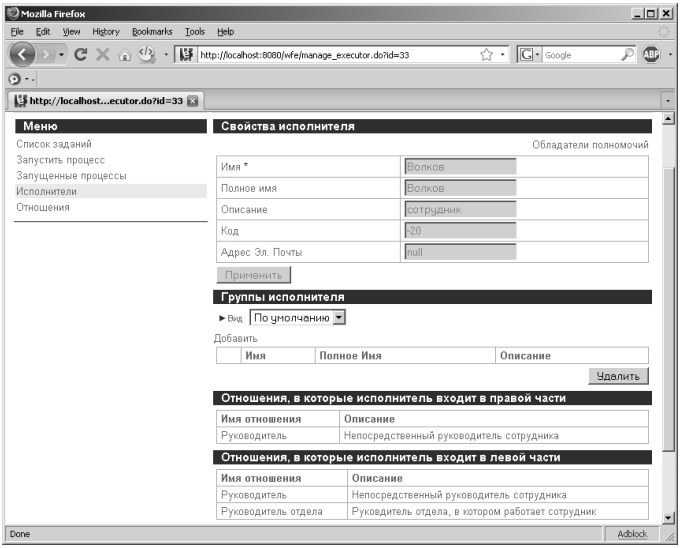
Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhost:8080/wfe/manage_executor.do?id=33
http://localhost...ecutor.do?id=33
Меню
Список заданий
Запустить процесс
Запущенные процессы
Исполнители
Отношения
Свойства исполнителя
Обладатели полномочий
Имя *
Волков
Полное имя
Волков
Описание
сотрудник
Код
20
Адрес Эл. Почты
null
Применить
Группы исполнителя
Вид
По умолчанию
Добавить
Имя
Полное Имя
Описание
Удалить
Отношения, в которые исполнитель входит в правой части
Имя отношения
Описание
Руководитель
Непосредственный руководитель сотрудника
Отношения, в которые исполнитель входит в левой части
Имя отношения
Описание
Руководитель
Непосредственный руководитель сотрудника
Руководитель отдела
Руководитель отдела, в котором работает сотрудник
Done
Adblock

Рис. 2.

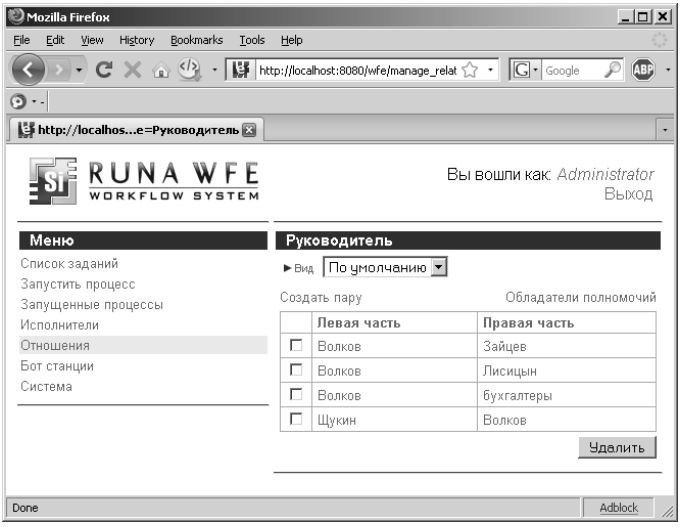
Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhost:8080/wfe/manage_relat
http://localhos...e=Руководитель
RUNA WFE
WORKFLOW SYSTEM
Вы вошли как: Administrator
Выход
Меню
Список заданий
Запустить процесс
Запущенные процессы
Исполнители
Отношения
Бот станции
Система
Руководитель
Вид
По умолчанию
Создать пару
Обладатели полномочий
Левая часть
Правая часть
Волков
Зайцев
Волков
Лисицын
Волков
бухгалтеры
Щукин
Волков
Удалить
Done
Adblock

Рис. 3.

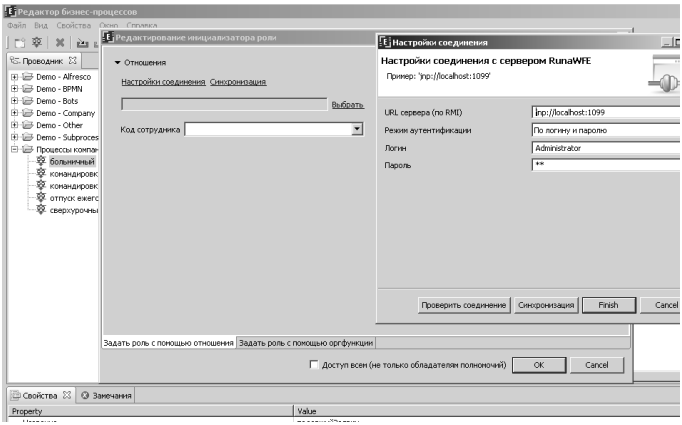

Рис. 4.

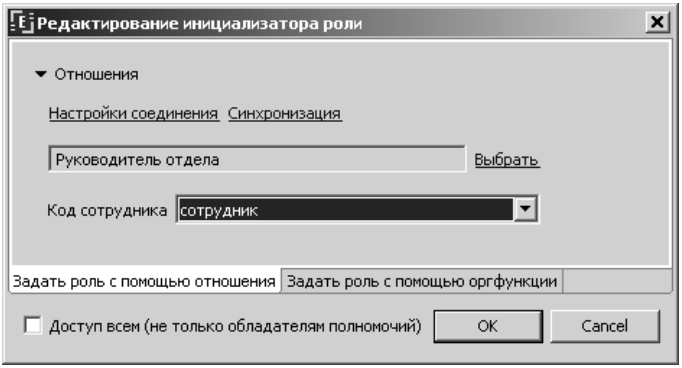

Рис. 5.

ния». В этом случае можно задать настройки соединения с сервером и импортировать отношения в редактор (см. Рис 4).

Далее отношение можно поставить в соответствие роли. В форме выбирается имя отношения и переменная или константа, соответствующая правой части отношения, задающая пользователя или группу пользователей (см. Рис 5).

Литература

[1] А. Н. Колмогоров, С. В. Фомин, Элементы теории функций и функционального анализа. 4 изд. М. Наука, 1976

[2] Ссылка на сайт проекта: `http://wf.runa.ru`

Антон Беляков, Александр Ковтушенко
Москва, МГТУ им. Н.Э. Баумана

Проект: bMPIdbg

Отладчик параллельных MPI программ — bMPIdbg

Аннотация

Отладчик bMPIdbg предназначен для работы с параллельными приложениями, использующими MPI. Предоставляет GUI к запускемым на узлах кластера сеансам gdbserver, поддерживает групповые команды и Python скрипты gdb.

Отладка распределенных и параллельных приложений качественно отличается от отладки последовательных программ. Параллельная программа выделяется тем, что ее поведение недетерминировано (выполнение отдельных ветвей параллельной прогаммы взаимно асинхронно). Отладка приложения в привычном режиме: точки останова, пошаговое выполнение; — принципиально не позволяет обнаружить многие ошибки. Параллельное приложение при каждом прогоне выполняется по иному, выполнение под управлением отладчиком существенно меняет прогон программы. Выявление специфических для параллельного счета ошибок требует отдельных усилий, данное приложение эти задачи в настоящее время не решает. Однако, подобная среда привычна программисту, существенно упрощает работу.

В данной области имеется передовой коммерческий продукт TotalView. Одним из принципиально важных возможностей является управление отладкой при помощи скриптов. Довольно удобно, когда средой отладки (ну, хотя бы, списком просматриваемых переменных) можно удобно манипулировать. При отладке параллельной программы, в которых выполняется последовательный счет возможно в нескольких тысячах ветвей, это не просто удобно, это критически необходимо.

Проект GDB с недавних пор поддерживает Python скриты, это создает возможность осуществить наш проект и может быть чрезвычайно полезным во многих случаях. По умолчанию, в ряде дистрибутивов эта опция отключена, требуется пересобрать gdb.

Наш отладчик состоит из следующих компонент:

- ParallelDebugger — графическая среда отладки:
- mpigdbserver — программа для запуска экземпляров gdbserver.

Запуск экземпляров gdbserver, запуск графической среды выполняется согласованно с системой управления кластером. Параллельный счет, как правило, выполняется в пакетном режиме у нас под управлением LoadLeveler. Для другого кластера потребуется переписать заново скрипты запуска задания.

Проект имеет информационные зависимости: Qt4.7.2, libmigdb.

Библиотека libmigdb предоставляет протокол GDB/MI (GNU DeBugger/Machine Interface), является проектом SourceForge(beta), упрощет написание графической среды отладки.

Литература

[1] Библиотека протокола GDB/MI, `http://sourceforge.net/projects/libmigdb/`

[2] `http://sourceware.org/gdb/current/onlinedocs/gdb/`

Иван Кутень
Беларусия/Минск,
Проект: Freedom Box `http://www.freedomboxfoundation.org`

Разработка Плаг-компьютера и «коробочка свободы»

Повышение производительности процессоров ARM архитектуры открыло новые области применения помимо традиционной мобильной электроники. Плаг-компьютеры — новая категория устройств, предназначенных для замены шумных и энергоемких x86 серверов

в домашних (personal) и SOHO применениях. Плаг-компьютер представляет собой мини-сервер, который включается прямо в розетку питания 220V, потребляет менее 5 ватт, и в тоже время имеет частоту ARM процессора от 800МГц до 2ГГц. В докладе приведен обзор программных решений и проблем возникших в ходе разработки устройства типа Плаг-компьютер. Далее в качестве одного из самых интересных применений таких Плаг-компьютеров мы рассмотрим проект FreedomBox и возможные технические решения на базе распределенных файловых систем, которые обеспечивают безопасное хранение персональных данных.

Михаил Якушин
Москва, ООО «Альт Линукс»

Проект: Sisyphus `http://www.altlinux.org/Ports/ARM`

Портирование Sisyphus на архитектуру ARM.

Аннотация

Доклад о том, что такое ARM, зачем он нужен и как мы на него переносили Sisyphus.

Архитектура ARM существовала давно, но только последние годы её вычислительной мощности стало хватать для запуска полноценных ОС. Сейчас возможно на этой архитектуре создавать не только мобильные устройства, но десктопы и мини-сервера. Поэтому появилась потребность переноса репозитария Sisyphus на эту архитектуру тоже. Для этого был проведен bootstrap, перенесены сборочные инструменты (gear, hasher), после чего была перенесена сборочная система girar-builder. Для автоматизации процесса были разработаны роботы, которые следили за основной частью сизифа и добавляли задания в сборочную систему для ARM.

Сейчас уже перенесено 8925 пакетов. Для упрощение сопровождения были разработаны дополнительные инструменты, позволяющие визуализировать состояние репозитария. Из-за особенностией архитектуры, ядро Linux приходиться собирать почти под каждую систему индивидуально, что усложняет тиражирование. Работы по данному направление продолжаются, репозитарий расширяется, и

уже сейчас существуют несколько сборок прошивок для конкретных устройств.

Дмитрий Кузьменко
Москва, ООО «Айбэйз»

Проект: Firebird `http://www.firebirdsql.org`

Опыт использования Firebird для критически важных приложений на предприятиях и в государственных организациях в России и за рубежом

За 10 лет существования СУБД Firebird распространилась очень широко, ее применяют в самых разнообразных системах — открытых, коммерческих или используемых в госструктурах — начиная от движка доступа к данным для справочных систем, распространяемых на компакт-дисках, до ERP-систем, обрабатывающих сотни гигабайт данных, и обслуживающих сотни пользователей. Какие требования к приложениям, СУБД и оборудованию предъявляют нынешние, все увеличивающиеся объемы данных?

Евгений Новиков, Алексей Хорошилов
Москва, Институт Системного Программирования РАН

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

Реализация аспектно-ориентированного программирования для языка Си

Аннотация

Доклад описывает подход к реализации парадигмы *аспектно-ориентированного программирования* (АОП) для языка Си. Основной упор при его разработке был сделан на особенностях языка Си, а также процесса сборки программ на этом языке, что отличает этот подход от более распространенной практики переноса концепций классической реализации АОП в AspectJ на другие языки программирования.

> Область практического применения предложенной реализации в проекте верификации драйверов Linux выходит за рамки традиционного АОП, что накладывает дополнительные требования на реализацию. В докладе рассматриваются эти требования, описывается состояние имеющейся реализации и намечаются дальнейшие пути развития инструмента.

Парадигма АОП нацелена на предоставление средств для повышения модульности программ за счёт выделения сквозной функциональности в виде так называемых аспектов. *Сквозная функциональность* — одно из основных понятий АОП — пронизывает основную функциональность программ, реализованную в виде функций, классов, модулей и т. п. В конечном итоге это усложняет программы, приводит к дублированию исходного кода и является источником ошибок различного рода. Типичными примерами сквозной функциональности являются: журналирование, обработка ошибок, работа с базами данных и т. д.

Для выделения сквозной функциональности в АОП используются специальные конструкции, называемые *аспектами*. Следующий пример демонстрирует основные конструкции АОП для AspectJ, самой известной реализации АОП для языка программирования Java [1]:

```
// Аспект.
aspect Logging {
  // Именованный срез задает соответствие
  // точкам соединения, вызовам методов.
  pointcut move():
    call(void FigureElement.setXY(int,int)) ||
    call(void Point.setX(int))              ||
    call(void Point.setY(int));
  // Рекомендация указывает действия, которые
  // необходимо выполнить непосредственно перед
  // выполнением соответствующей срезу точки программы.
  before(): move() {
    System.out.println("about to move");
  }
}
```

Помимо того, что реализация АОП должна позволять выделять сквозную функциональность в виде аспектов, она также должна ав-

томатически связывать аспекты с каким-либо представлением программ. Данный процесс называется *инструментированием*. Более подробно можно посмотреть, например, [2].

По приведенному выше примеру видно, что реализация АОП сильно зависит от языка программирования. Поэтому, при реализации АОП для языка программирования Си, мы учли особенности данного языка (например, наличие макродиректив и адресной арифметики), а также особенности процесса сборки программ (три стадии: препроцессирование, компиляция и компонование). В соответствии с этим мы предложили подходящий набор конструкций АОП и определили процесс инструментирования особым образом.

Инструментирование в предложенном подходе проходит в четыре стадии, на каждой из которых вызывается модифицированный компилятор *LLVM GCC Front End*. На первой стадии, в соответствии с заданными срезами и рекомендациями, до или после исходного кода обрабатываемого Си файла дописывается заданный пользователем код, например, декларации вспомогательных функций и включения заголовочных файлов. На второй стадии производится *макроинструментирование*, благодаря которому можно заменять исходные макроподстановки на текст, заданный пользователем. Также, на второй стадии производится стандартное препроцессирование. Третья и четвертая стадии соответствуют этапу компиляции файла. При этом, на основе срезов и рекомендаций, генерируются определения вспомогательных функций, которые выполняют требуемые действия. Эти функции дописываются напрямую в конец файла. На четвертой стадии происходит конечная компиляция файла, одновременно с которой исходные конструкции программы связываются с вспомогательными функциями. В результате на выходе получается инструментированный объектный файл. Следует отметить, что при выполнении инструментирования автоматически за счёт средств компилятора проверяется корректность аспектов. Благодаря тому, что инструментирование выполняется на всех этапах сборки программы, подход является достаточно полным и универсальным в том смысле, что при необходимости его достаточно легко расширить. Подробнее см. [3].

При разработке подхода существенное внимание было уделено его практическому применению. В настоящее время, предложенная реализация, в основном, используется для задания правил корректности [4] в виде аспектов и последующего инструментирования исходного кода ядра Linux в рамках проекта верификации драйверов Linux

(LDV) [5]. Ввиду того, что используемые инструменты верификации работают с исходным кодом на языке Си, то объектный код, получаемый после применения аспектов, трансформируется обратно в код на языке Си. Для этого используются средства LLVM: во-первых, происходит компоновка требуемых инструментированных объектных файлов, а затем генерация исходного кода.

Результаты практического применения показали, что подход в целом себя оправдывает. Из 22 различных типов правил корректности LDV, проанализированных на настоящий момент, с помощью разработанных инструментов можно формализовать 14. Оставшиеся типы могут быть формализованы после реализации определения срезов с участием специфичных конструкций, таких как, взятие поля переменной типа структуры, разыменование указателя определенного типа и др. На уже формализованных правилах применение инструментов верификации позволило выявить несколько десятков серьезных ошибок в коде драйверов

Linux, о которых было сообщено разработчикам и которые были исправлены [6].

В рамках применения инструментирования в проекте по верификации драйверов проявился существенный недостаток предложенного подхода, который не является проблематичным при традиционном использовании АОП для генерации машинного кода: исходный код, получаемый после инструментирования, значительно отличается от первоначального представления, что создает определенные трудности как для статического анализатора, так и для последующего анализа результатов верификации пользователем. Преодоление данной проблемы в рамках инфраструктуры *LLVM GCC Front End* невозможно, поэтому наиболее оптимальным решением является реализация инструментирования непосредственно на основе компилятора gcc.

Таким образом, среди основных направлений дальнейшего развития можно выделить: 1) улучшение генерации исходного кода (в настоящее время ведется активная разработка генератора Си кода по внутреннему представлению GCC); 2) реализация специфичных для Си конструкций аспектно-ориентированного программирования (например, взятие поля переменной типа структуры).

Литература

[1] Пример из документации AspectJ, `http://eclipse.org/aspectj/doc/`

released/progguide/starting-aspectj.html

[2] Новиков Е.М., Мутилин В.С., Хорошилов А.В., Классификация концепций для аспектно-ориентированного расширения языка программирования C, Труды 52-й научной конференции МФТИ «Современные проблемы фундаментальных и прикладных наук», часть 7 «Управление и прикладная математика», 31-33, Москва-Долгопрудный, 27-30 ноября, 2009

[3] Novikov E., One Approach to Aspect-Oriented Programming Implementation for the C Programming Language, Proceedings of the 5th Spring/Summer Young Researchers' Colloquium on Software Engineering (SYRCoSE 2011), 74-81, Yekaterinburg, Russia, May 12-13, 2011

[4] Мутилин В.С., Хорошилов А.В., База правил для верификации драйверов Linux, Тезисы докладов 6-й конференции разработчиков свободных программ на ПРОТВЕ, 22, Обнинск, 27-28 июля, 2009

[5] Khoroshilov A., Mutilin V., Novikov E., Shved P., Strakh A., Towards an Open Framework for C Verification Tools Benchmarking, Ershov Informatics Conference (PSI'11), 82-91, Novosibirsk, Russia, June 27 - July 1, 2011

[6] Список обнаруженных ошибок в драйверах Linux, http://www.linuxtesting.ru/results/ldv

Владимир Дёмин, Дмитрий Костюк, Александр Никонюк
Брест, Брестский государственный технический университет

Расширение функционала оконного менеджера Compiz на основе средств нелинейного и дискретного масштабирования

Аннотация

Рассмотрены модели нелинейного и дискретного масштабирования окон и их практическая реализация в форме модуля расширения оконного менеджера Compiz версии 0.9x. Предлагаемые модули предназначены для уменьшения размеров окон за счет динамического сжатия части окна, прилегающей к краю рабочей области, а также для повышения наглядности работы с большим числом приложений за счет применения специализированной док-панели, объединяющей механизм виртуальных столов и концепцию мини-окон.

Из-за ограниченности аппаратных ресурсов, не позволяющих задействовать большие площади для вывода информации, пользователь часто не имеет возможности видеть одновременно изображение всего рабочего пространства. В последнее время проблема получила дополнительное развитие из-за роста популярности портативных устройств, способных запускать приложения, интерфейс которых изначально рассчитан на стандартное разрешение экрана. Подобные устройства оказываются не только не способны разместить на экране нужное количество окон, но зачастую не могут показать целиком одно стандартное окно.

Перспективный подход решения проблемы — применение уменьшенных изображений окон (либо их частей), не находящихся в фокусе работы пользователя. Современные аппаратно-ускоренные оконные менеджеры без дополнительной нагрузки на центральный процессор обеспечивают динамичность обновления миниатюр (т. н. live previews), позволяя естественным образом отслеживать все процессы, происходящие в многооконной системе.

В 2006 г. одним из авторов предложены две модели, находящиеся в русле данного подхода [1]. Модель дискретного масштабирования представляет собой обобщенный случай интерфейса на основе мини-окон и предполагает выделение на периферии рабочей области фиксированной зоны для отображения окон в уменьшенном масштабе, пропорциональном расстоянию от центра экрана до начала координат окна. Модель нелинейного масштабирования предполагает разделение рабочей области окна на центральную (1) и периферийную (2) зоны. В зоне 1 отображается основная информация окна, представленная в единичном масштабе. В зоне 2 выводится изображение, имеющее переменный масштаб, т.е. пиксели со смещением (функцией пространственных искажений), монотонно возрастающим вдоль координаты.

В настоящей работе представлена реализация моделей дискретного масштабирования (в варианте мини-окон) и нелинейного масштабирования на базе оконного менеджера Compiz 9.x. Разработка может быть достаточно просто перенесена на любой оконный менеджер, позволяющий совершать манипуляции с изображениями окон средствами аппаратно-ускоренной графики OpenGL.

Модуль dock, реализующий модель мини-окон, выстраивает миниатюризированные окна вдоль границы экрана, формируя аналог док-панели. По умолчанию миниатюры располагаются стандартно,

но могут быть перетаскиванием объединены в плотно расположенные группы. Неминимизированные окна, расположенные на экране, также составляют отдельную группу, не отображаемую в панели мини-окон. Нажатие горячей клавиши (например, сочетания Win+Tab) вызывает переключение групп между отображением в единичном масштабе на стандартных позициях и отображением в в панели мини-окон, объединяя функционал переключателя задач и пейджера. Кроме того, группирование функционально с точки зрения группового запуска приложений из окна истории групп [2].

Модуль twist, реализующий нелинейное масштабирование, активируется при перетаскивании окна за пределы рабочей области (например, границы экрана). Сжатие соответствующей части окна отображается в реальном времени в процессе перетаскивания, и т. о. пользователь имеет возможность регулировать коэффициент сжатия для достижения баланса между размерами и читаемостью содержимого.

Для доступа к текстуре скрытой части окна модуль предварительно выполняет масштабирование с коэффициентом, близким к единице, после чего создает копию части текстуры, соответствующей сжимаемой части. Дублированное изображение выводится поверх видимого фрагмента окна [3].

В обеих реализациях изменение масштаба окна затрагивает только его изображение. Для самого окна при этом не происходит никаких изменений в размерах, иначе была бы нарушена обратная совместимость с существующими приложениями. Из экономии вычислительных ресурсов вместо мониторинга и пересчета координат указателя мыши выполняется блокирование соответствующих событий для сжатой области, а события клавиатуры остаются доступны, что позволяет взаимодействовать с рядом приложений. Подход также хорошо сочетается с группированием меню и панелей в единый блок, присутствующим в типовых интерфейсах.

В итоге разработанные модули позволяют добиться более экономного использования площади экрана без модификации существующего программного обеспечения.

Литература

[1] *Борушко И.Н., Гоманова Е.В., Костюк Д.А.* Применение модели периферического зрения в графическом интерфейсе пользователя. // Совре-

менные информационные компьютерные технологи: сб. науч. ст. Гродно: ГрГУ, 2006. Стр. 22–27

[2] *Костюк Д.А., Дёмин В.В.* Модель мини-окон с динамическим отображением в аппаратно-ускоренном графическом интерфейсе // Вестник БрГТУ. — 2009, №5: Физика, математика, информатика. — С. 71–74.

[3] *Никонюк А.Н.* Нелинейное масштабирование окон для экономии площади дисплея портативных устройств // Сучасні проблеми радіотехніки та телекомунікацій «РТ — 2011»: Матер. 7-ої міжнар. молодіжн. наук.-техн. конф., Севастополь 11–15 квітня 2011 р. — Севастополь: СевНТУ, 2011. — С. 362.

Игорь Воронин
Московская обл./Шатура, ИПЛИТ РАН Шатура
http://wsn.laser.ru

Проблемы адаптации распределенной сенсорной сети для управления устройствами и агрегатами

Аннотация

Распределенные сенсорные сети на основе свободного программного обеспечения могут быть использованы для управления большим числом устройств и механизмов из единого центра. В докладе описывается решение проблем синхронизации циклов сна и времени хранения буфера команд для управления агрегатами с высокой надежностью.

Проведение экспериментальных исследований с распределенными сенсорными сетями (РСС) позволило не только производить экологический мониторинг различных объектов например замеряя параметры температуры, давления, влажности, освещенности, так и подтолкнуло к необходимости производить управление устройствами для устранения различных проблем — например пожары, или для передвижения детекторов автоматически измеряющих радиацию в точках замеров и самостоятельно покидающих опасную зону. Использование РСС для управления устройствами все шире внедряется в повседневную действительность.

Необходимо учитывать, что программный стек — прошивка для управления устройствами — как правило задается заводом поставщиком микрочипов zigbee, и, как правило, не поддается изменению

сторонним разработчикам. Но при этом необходимость гарантированности доставки пакетов данных и выработка рекомендаций по сбору информации с узлов на центральный коллектор остается актуальной.

Надо также учитывать тот факт, что для экономии энергии — конечное устройство оснащеное атономным источником питания находится в режиме периодического сна (пониженого расхода энергии), а центральный узел, рассылающий запросы и принимающий ответные пакеты данных — имеет ограниченный объем хранимого буфера пакетов и может возникать ситуация, когда стоящий в очереди пакет данных — замещается следующим, а узел сети РСС, которому он был адресован, еще не проснулся.

Для исследований и проведения экспериментов над РСС был выбран коммерчески доступный робот, набор-конструктор, который после некоторой модернизации он был оснащен модулем РСС, что в себя включало:

1. Необходимое оборудование: комплектующие для сборки роботов, наборы датчиков, сетевое оборудование.
2. Разработанное программное обеспечение для управления робототехническим комплексом.
3. Созданные технологические схемы управления механизмом робота-конструктора.
4. Составлены диаграммы сна и переполнения буфера в различных вариациях.
    - система команд робота строго определена. Информация вводится с помощью датчиков, выводится через линии связи с другими устройствами;
    - программа для робота состоит из отдельных команд;
    - робот исполняет предложенную ему программу;
    - поведение робота зависит от качества программы, а значит и от опыта самого программиста, если робот делает что-то не так, как задумывалось — происходит коррекция программы.

Для управления роботом было использовано свободное ПО AltLinux 5.0 Gnome, редактор Geany, gcc4.2, для визуализации использованы библиотеки QT3.

В результате проведенных экспериментов выснилось, что как говорилось выше, для экономии энергии в РСС конечные модули имеют

автономные источники питания и поэтому с некотором периодом входят в режим сна, а если дается команда движения вперед и эта команда достаточно долго — например более 30 сек остается постоянной, иначе говоря не поступает ни какой другой команды за этот промежуток времени, то машинка-конструктор движется вперед и при этом микрочип переходит автоматически в режим «сна». В тоже время, когда наступает необходимость выполнить какой-то маневр, и приходит команда повернуть или остановиться, то машинка еще некторое время продолжает движение вперед — пока не наступит периодически ожидаемое «пробуждение» чип проснется и обработает команду поворота или остановки.

Подобная ситуация рассогласования времени сна устройства и времени подачи команды на маневр, конечно же недопустимо для управления устройствами в реальном времени. Поэтому были выработаны алгоритмы сна устройствами — в тот момент когда происходит движение механизмов. В режиме покоя сон может быть организован по алгоритму заданному поставщиком микрочипа.

Используемые для экспериментов с РСС наборы роботов-конструкторов могут использоваться, при соответствующем методическом обеспечении учебными материалами, так же и в школьных занятиях дополнительного образования, например, на уроках труда. Тем самым для интересующихся учеников буден доступен недорогой прибор, который они смогут исследовать, поизучать, как говориться, на кончиках пальцев, прикоснуться к инновационному техническому творчеству, а вместе с тем пофантазировать и создать на этой основе СПО собственную разработку для использования в различных научно-практических конференциях проходящих в школах ко дню науки в начале февраля.

Георгий Курячий
Москва, ALT Linux
Проект: Sisyphus http://sisyphus.ru

Собирается ли свободное сообщество использовать для обновления своих пакетов огромных управляемых человекоподобных роботов?

Обновление уже оформленного пакета в хранилище (например, при выходе новой upstream-версии ПО) — сравнительно несложная задача, посильная как человеку, так и небольшому роботу. Массовое отслеживание новых версий и обновление соответствующих пакетов в хранилище — задача, очевидно, требующая более сложных и больших роботов, отчасти похожих на участников сообщества, а отчасти — управляемых ими. В докладе обсуждаются наиболее востребованные свободным сообществом свойства огромных управляемых человекоподобных роботов, приводятся практические примеры.

Денис Силаков
Москва, ИСП РАН
Проект: Центр верификации ОС Linux http://linuxtesting.ru

Контроль доступа приложений к сетевым ресурсам в условиях недоверенной ОС

Аннотация

Доклад посвящен системе безопасности, обеспечивающей контроль взаимодействия приложений с удаленными компьютерами по сети. Архитектура системы основана на использовании технологии аппаратной виртуализации. Предполагается, что операционная система не является доверенной и содержит уязвимости и закладки, которые могут быть использованы злоумышленниками для перехвата или искажения конфиденциальной информации.

Современные ОС и вопросы безопасности

В современных вычислительных системах конфиденциальность и целостность пользовательских данных зависит не только от корректности работы прикладного программного обеспечения, которому санкционирован доступ к этой информации, но и от корректности операционной системы. В ряде случаев пользователь может в недостаточной мере доверять операционной системе, чтобы выполнять обработку данных на данном компьютере без риска для их безопасности. Во-первых, такая ситуация может возникнуть в случае отсутствия у пользователя возможности проконтролировать конфигурацию системного программного обеспечения, установленного на компьютере (например, при использовании общедоступного компьютера или при выполнении вычислений в облаке). Во-вторых, недостаточный уровень доверия к операционной системе может быть обусловлен повышенными требованиями к безопасности обрабатываемых данных.

Повышение уровня доверия к вычислительной системе может достигаться путем сокращения числа компонент, имеющих неограниченный (неконтролируемый) доступ к защищаемым данным. В частности, из числа таких компонент можно исключить операционную систему. Средства самой ОС для такой цели не подходят; одним из возможных решений задачи является использование аппаратной виртуализации, поддерживаемой многими современных процессорами. Именно такой подход используется в системе безопасности, разрабатываемой в ИСП РАН.

Система безопасности, основанная на использовании гипервизора

В предлагаемом подходе, операционная система и приложения выполняются внутри виртуальной машины, а система безопасности реализуется в теле монитора виртуальных машин (гипервизора). Гипервизор аппаратно защищен от несанкционированного доступа со стороны кода в виртуальной машине и в то же время имеет полный доступ к ее ресурсам. В настоящее время используется гипервизор, основанный на KVM.

В системе безопасности используются две ВМ — основная, в которой работает пользователь, и сервисная, используемая гипервизором для служебных целей. Основная ВМ не имеет сетевого интерфейса и

при попытке доступа к удаленным машинам ядро ОС будет сообщать приложениям о недоступности сети. Однако для ряда приложений, считающихся доверенными, гипервизор предоставляет возможность такого доступа. Для этого производится перехват системных вызовов, поступающих в ядро основной ОС, и в случае, если вызов, относящийся к сетевому взаимодействию, поступил от доверенного приложения, этот вызов передается на обслуживание в сервисную ВМ. У сервисной ВМ есть сетевой интерфейс, с помощью которого и производится обмен данными с удаленными машинами.

Для упрощения контроля процессов, работающих внутри ОС, основной ВМ предоставляется в распоряжение одноядерный процессор (так что в любой момент времени внутри ВМ работает только один процесс).

Для предотвращения вредоносного воздействия на код доверенных процессов, система безопасности контролирует целостность выполнения приложений внутри ВМ. Гарантируется обнаружение несанкционированной модификации потока управления и адресного пространства процесса (в том числе выполняемой из контекста ядра операционной системы) в каждый момент времени начиная от загрузки и до завершения приложения. При этом система безопасности учитывает все особенности реализации современных операционных систем, включая механизм подкачки и прямой доступ к памяти (DMA).

Заключение

Предложенная система безопасности позволяет частично или полностью (в зависимости от требований безопасности) исключить операционную систему из доверенной вычислительной базы приложения. При этом в доверенную вычислительную базу добавляется гипервизор. Однако аппаратная поддержка технологии виртуализации позволяет реализовать гипервизор с суммарным размером кода в среднем на три порядка меньше, чем ядро современных массовых операционных систем. В итоге значительное сокращение размера доверенной вычислительной базы обеспечивает повышение доверия к вычислительной системе в целом.

В настоящее время разработанный прототип позволяет контролировать доступ приложений к сети, однако используемый подход может быть расширен для контроля доступа к произвольным аппарат-

ным ресурсам (например, жестким дискам и другим устройствам хранения информации).

Игорь Власенко
Украина/Киев, ALT Linux

Проект: Автосопровождение пакетов
`http://www.altlinux.org/Gear/cronbuild`

Автоматизация сборки и сопровождения пакетов для дистрибутива.

Когда-то сборка и пересборка исходного пакета была достаточно сложным и кропотливым занятием, отнимавшим заметную долю времени при сопровождении пакета. Сейчас эти задачи автоматизированы настолько, что субъективно воспринимаются как атомарные операции. Репозиторий СПО “Sisyphus” может заслуженно гордиться, наверное, лучшей в своем классе системой сборки hasher, а также своим автоматизированным incoming.

В то же время обновление исходного пакета при выходе новой версии у нас в ALTLinux Team все еще является традиционно ручной задачей, оттягивающей на себя значительную часть времени при сопровождении пакетов. Это слишком расточительно: участников ALTLinux Team не так много по сравнению с числом исходных пакетов, что приводит к тому, что много пакетов недостаточно хорошо сопровождаются: они отстают по версиям, годами не исправляются простые ошибки.

В докладе будет рассказано об успешных опытах по автоматизации сопровождения исходных пакетов — проектах autoports и fcimport, а также сопутствующей им инфраструктуре. Проект autoports во время испытательного срока собрал более 2000 пакетов для репозитория 5.1. В рамках проекта fcimport сейчас сопровождается более 400 пакетов репозитория СПО “Sisyphus”, при чем это только тестовое множество, используемое при разработке fcimport.

Цель этих экспериментов — исследовать возможность автоматизации сопровождения пакета. hasher у нас полностью автоматизировал

процесс сборки пакета. сопровождение пакета — более сложная задача, но прогресс в этом направлении позволяет освободить майнтайнера от механических работ, чтобы у него было больше времени для высокоуровневых задач, таких, как исправление ошибок и подгонка под стандарты качества дистрибутива.

В отличие от hasher, задачи подготовки и обновления исходного пакета невозможно полностью автоматизировать из-за того, что сами репозитории программного обеспечения, политики упаковки и инструменты сборки находятся в процессе постоянного изменения. Поэтому автоматизированная сборка никогда не сможет полностью вытеснить ручную. Для нее достаточно, чтобы она была статистически успешной с хорошей вероятностью, чтобы ручные работы выполнялись только тогда, когда они действительно необходимы.

Михаил Шигорин
Украина/Киев, Massive Solutions Ltd, ALT Linux Team
Проект: Sisyphus `http://sisyphus.ru`

Скрестим вилки — fork? merge!

Скрестим вилки

Доступность исходного кода может подталкивать к появлению множественных веток развития. Такая фрагментация опасна распылением усилий, появлением несовместимостей, застоем разработки — но грамотное сведение разнообразных веток разработки в единое целое способно мощнейшим образом способствовать разносторонности функций.

- что такое форк и мерж
- цена отличия и слияния
- совместная конкуренция

Merge-a-fork

Source code availability might tempt to “just make a copy of it”. This might lead to fragmentation which in its turn is capable of splintering the efforts, introducing incompatibilities, and even development stall — but doing it properly may also heavily help with the feature diversity.

- what's a fork and a merge
- the price of divergence and convergence
- joint competition

Когда мы что-то делаем, то нередко основываемся на уже существующем и дополняем или дорабатываем его. При этом вне зависимости от того, код это, документация или конфигурация — происходит фактическое «раздвоение» объекта. И если уже существующее продолжает развиваться, то это раздвоение называется «форк». Если такие производные варианты сливаются полностью или частично, постоянно или периодически — такое объединение называется «мерж».

Форк чреват тем, что либо усилия на развитие в основе схожих вещей будут дублироваться (без малого худший случай), либо потребуются дополнительные решимость, время и силы на периодическое «втягивание» наработок коллег, либо же произойдёт загнивание менее продуктивной ветки вместе с уникальными для неё наработками.

С другой стороны, ожидаемый форк (когда событие ответвления с целью дальнейшей разработки является ожидаемым и приветствуемым) может оказаться весьма мощным средством проверки самых различных идей реализацией: тогда наиболее удачные затем прямо или же после переработки (как правило, с учётом возникших замечаний коллег и параллельно происшедших изменений в затрагиваемой части) мержатся в более майнстримную ветку.

Мерж чреват в первую очередь неизбежной затратой времени — как на собственно техническую часть вопроса, так и на предварительное согласование с последующей утруской и усушкой вновь образовавшихся проблем.

Хорош же мерж тем, что уменьшение объёма разницы между развивающимися параллельно ветками приводит к уменьшению латентных затрат времени на отслеживание и втягивание интересных наработок коллег.

В докладе рассматриваются типичные виды, причины и последствия форков (причём всегда есть что добавить по опыту аудитории), а также более-менее соответствующие варианты мержей.

Григорий Злобин
Украина/Львов, Львовский национальный университет им. И. Франко

Использование свободного программного обеспечения в учебных заведениях Украины: попытка анализа

Несмотря на положительный опыт использования свободного программного обеспечения (СПО) в образовании как в странах ближнего, так и далекого зарубежья в Украине доселе не принята концепция использования СПО в образовании. Вместе с тем усилиями энтузиастов в учреждениях образования Украины свободное программное обеспечение все же используется! Через отстраненную позицию министерства образования и науки Украины нет исчерпывающей информации об опыте использования СПО в образовании. Благодаря тому, что во Львовском национальном университете имени Ивана Франко 1-6 февраля 2011 г. состоялась довольно представительная международная научно-практическая конференция “FOSS Lviv-2011”, появилась возможность проведения анализа использования СПО в высших учебных заведениях Украины. Из 81 докладов, поданных на конференцию, 39 посвящены использованию СПО в учебных заведениях. Эти доклады можна сгруппировать за такими направлениями (названия докладов поданы на языке оригинала):

I. Дистанционное обучение

1. Артеменко В.Б. «Розроблення електронного деканату для системи управління дистанційним навчання MOODLE» Львовская коммерческая академия

2. Коцаренко М.В., Бойко О.В. «Вибір платформи дистанційного навчання» Львовский национальный медицинский университет им. Данила Галицкого

3. Макаренко І.Є., Мерзлікін П.В. «Використання контрольно-діагностичної програми iTest у ході моніторингу якості процесу навчання старшокласників» Криворожский государственный педагогический университет

4. Маркова Є.С. «Використання системи Moodle для організації контролю знань майбутніми вчителями-гуманітаріями» Бердянский государственный педагогический университет

5. Сергієнко В.П., Сліпухіна І.А. «Тестування в Moodle як елемент менеджменту якості освіти: перший досвід» НПУ им. М.П. Драгоманова

6. Жарких Ю.С., Лисоченко С.В., Сусь Б.Б., Третяк О.В. «Особливості програмного забезпечення в електронному навчанні» Киевский национальный университет имени Тараса Шевченко

7. Триус Ю.В. «Інформаційно-аналітична система управління навчальним процесом ВНЗ на базі Moodle» Черкасский государственный технологический университет

8. Франчук В.М. «Використання CMS JOMLA! та LCMS MOODLE у ВНЗ» НПУ им. М.П. Драгоманова

9. Франчук В.М. «Локалізація системи MOODLE» НПУ им. М.П. Драгоманова

10. Захарченко В.М., Шапо В.М. «Застосування вільного програмного забезпечення для дистанційного навчання у вищих навчальних закладах» Одесская национальная морская академия

II. Использование систем компьютерной математики

1. Бугаєць Н.О. «Використання вільно-поширюваного ПЗ математичного призначення в університеті» НПУ им. М.П. Драгоманова

2. Лазурчак І.І., Кобильник Т.П. «Вільнопоширювані системи комп'ютерної математики в освіті і науці» Дрогобычский государственный университет имени Ивана Франко

3. Лов'янова І.В. «Використання комп'терних математичних систем у професійній підготовці майбутнього вчителя математики» Криворожский государственный педагогический университет

4. Філь І.М. «Моделювання задач електротехніки у XCOS» Донецкий национальный технический университет

5. Чичкарьов Є.А. «Розробка і використання web-інтерфейсів для роботи з системами комп'ютерної математики» Приазовский государственный технический университет

6. Яхненко І.В., Лутфулін М.В. «Про комп'ютерний супровід викладання геометрії» Полтавский национальный педагогический университет им. В.Г. Короленко

III. Общие вопросы использования СПО в образовании

1. Апуневич С.Є., Злобін Г.Г., Рикалюк Р.Є., Шувар Р.Я. «Використання вільного програмного забезпечення в навчанні і наукових дослідженнях у Львівському національному університеті імені Івана Франка» Львовский национальный университет имени Ивана Франко

2. Воронкін О.С.«Використання вільного програмного забезпечення в системі дистанційної освіти» Луганский государственный институт культуры и искусства

3. Єфименко В.В. Вільнопоширюване програмне забезпечення курсу «Нові інформаційні технології» для студентів спеціальності «Біологія» НПУ им. М.П. Драгоманова

4. Покришень Д.А., Дрозд О.П., Сподаренко І.Й. «Використання вільного програмного забез-печення у професійній підготовці майбутніх інженерів» Черниговский государственный технологический университет

5. Риковський П.А. «Про досвід використання ОС Linux у навчальному процесі Львівського національного медичного університету імені Данила Галицького» Львовский национальный медицинский университет им. Данила Галицкого

6. Спірін О.М., Сверчевська О.С. «LINUX та VIRTUAL-BOX у навчанні абстрактних понять теорії операційних систем» Житомирский государственный университет имени Ивана Франко

8. Харченко В.М. «З досвіду використання вільного програмного забезпечення при вивченні інформатики» Нежинский государственный университет им. Николая Гоголя

IV. Использование открытых средств программирования

1. Коркуна Т.Й. «Використання відкритих програмних засобів в процесі навчання статистич-ним дисциплінам» Самборский техникум экономики и информатики

2. Костюк Д.А. «Построение практикумов по программированию и архитектуре ЭВМ на базе GNU/LINUX» Брестский государственный технический университет

3. Мелех Б.Я., Тишко Н.Л., Коритко Р.І. «Розрахунок фотоіонізаційних моделей небулярного газу в ОС LINUX UBUNTU 10.10 та WINDOWS 7» Львовский национальный университет имени Ивана Франко

4. Шийка Ю.Я., Шувар Р.Я. «Реалізація розподілених обчислень на основі грід-платформи з відкритим кодом BOINC» Львовский национальный университет имени Ивана Франко

5. Шувар Р.Я., Бойко Я.В. «Реалізація високопродуктивної обчислювальної системи на базі ОС LINUX» Львовский национальный университет имени Ивана Франко

V. Разработка программного обеспечения

1. Мартинюк-Лотоцький К.П., Білінський А.І. «Комплекс програм для лазерних спосте-режень штучних супутників Землі» Львовский национальный университет имени Ивана Франко

2. Сподарець Д.В., Драган Г.С. «Розробка системи спектральної діагностики димової плазми» Одеский национальный университет имени И.И. Мечникова

3. Злобін Г., Скляр В., Чмихало О., Шевчик В. «Використання вільного програмного забез-печення для створення програми керування інформаційним автоматом» Львовский национальный университет имени Ивана Франко

4. Малихіна Т.В. «Використання бібліотеки класів GEANT4 в ОС Linux при розробці програмного забезпечення для моделювання процесів взаємодії випромінювання з речовиною» Харьковский национальный университет имени В.Н. Каразина

VI. Отдельные доклады

1. Гриценко В.Г. «Інформаційна технологія управління навчальним навантаженням у вищих навчальних закладах» Черкасский национальный университет имени Б. Хмельницкого

2. Злобін Г.Г. «Про досвід використання офісного пакету OpenOffice.org.ukr в курсі інформатики для економічних і юридичних спеціальностей ВЗО» Львовский национальный университет имени Ивана Франко

3. Матвієнко Ю.С. Досвід використання редактора Gimp при вивченні курсу «Комп'ютерна графіка і дизайн» Полтавский национальный педагогический университет им. В.Г. Короленко

Таким образом можно констатировать как широкий спектр использования СПО в украинских учебных заведениях от дистанционного обучения до разработки открытого програмного обеспечения, так и широкую географию использования СПО от Луганска на во-

Рис. 1.

стоке до Львова на западе и от Чернигова на севере до Одессы на юге (см. рис.1).

Литература

[1] Тези міжнародної науково-практичної конференції FOSS LVIV-2011. Збірник наукових праць /за ред. Злобіна Г.Г., Апуневича С.Є., Машкова В.В., Апуневич С.В. Вид-во ЛНУ імені Івана Франка. 2011 — 196 с.