# О проекте

## Что такое osFree?

osFree это проект по созданию открытой операционной системы, основанной на микроядре L4, целью которого является бинарная совместимость с OS/2 (IA32). Также, возможно параллельное сосуществование API различных операционных систем, построенных поверх одного и того же микроядра.

## Зачем нужно микроядро?

- микроядро может служить базой для параллельных API, реализованных поверх него. Эти API могут существовать независимо, имея минимальную общую базу, состоящую из самого микроядра и некоторого набора общих сервисов (называемых personality-нейтральными сервисами). Это позволяет иметь независимую (non-layered) реализацию параллельных API.
- API ОС поверх микроядра реализуется полностью в пространстве пользователя, оставляя только микроядро в режиме ядра. Такие компоненты ОС как планировщики процессов, менеджеры памяти, процесс подкачки, и даже компоненты, осуществляющие прямой доступ к оборудованию и обработку прерываний, также перемещены в пространство пользователя.
- это позволяет создать устойчивую систему с очень стабильным отлаженным крошечным ядром и менее стабильными компонентами из пространства пользователя, которые (как показывает, например, дизайн Minix) могут быть перезапущены (причем, даже автоматически) после того, как возникла неисправность. Само микроядро может быть очень отлажено, и даже верифицировано формальными методами на отсутствие ошибок. Поэтому ошибка в микроядре маловероятна, а другие компоненты могут быть не столь критичными.
- это также позволяет использовать обычные (“прикладные”) технологии разработки для драйверов, так как драйвер в микроядерной системе не отличается кардинально от обычных приложений.
- микроядерная архитектура также улучшает структурированность системы, то есть, зависимости между ее компонентами более четкие. Серверы работают поверх микроядра, взаимодействуют только через жестко заданные интерфейсы и скрывают свое внутреннее устройство, что имеет четкие параллели с объектно-ориентированным подходом.
- микроядерная система улучшает изоляцию ошибок внутри компонентов системы, так как все серверы исполняются в отдельных адресных пространствах.
- микроядро абстрагирует аппаратные интерфейсы от серверов пространства пользователя, что дает возможности для создания переносимых операционных систем, в которых все компоненты пространства пользователя остаются неизменными (на уровне исходного кода), необходима только их рекомпиляция.

## Почему L4?

- L4 это микроядро второго поколения, значительно улучшающее общую производительность системы. Это может быть наглядно заметно на примере l4linux.

L4linux – это, в сущности, порт обычного ядра Linux на новую архитектуру “l4”. Его код, напрямую работающий с оборудованием, изменен таким образом, что оно становится доступно не напрямую, а через механизмы L4. Производительность ядра Linux может быть оценена с помощью тестов (benchmarks), которые показывают потерю всего около 2% производительности, по сравнению с “родным” ядром Linux. Мы пробовали запускать l4linux на реальной машине и на глаз не заметили никакого различия в производительности. Это показывает, что L4 дает замечательную производительность с очень маленькими накладными расходами.

- Его минимальность и перемещение всей логики работы (policies) вне ядра, с оставлением внутри него только минимального набора механизмов, делает его почти универсальным и позволяет реализовать практически любое нужное API.
- Мы не хотим изобретать велосипед, и L4 содержит практически все нужные нам механизмы.
- Для L4 уже реализованы наборы готовых сервисов общего назначения, то есть, у нас уже есть готовый “конструктор”, а не только голое ядро.
- В качестве базы для Linux personality можно использовать уже готовый l4linux. Он все еще в стадии отладки, но почти все, тем не менее, работает. На нашем ноутбуке, только PCMCIA модем не заработал, из всего оборудования (это можно, впрочем, исправить, сделав ioremap на другие адреса – исходные адреса, используемые драйвером, заняты в L4-based системе). Wi-fi, bluetooth стек, USB стек, файловые системы, запись CDROM – все это заработало! Да, еще одно неудобство: так как видео работает поверх родной графической L4 консоли, которая на данный момент работает в режиме VESA (только избранные видеокарты поддерживаются с акселерацией), то поддержка видео, конечно, ограниченная.
- Device Driver Environment (DDE) может быть использовано в будущем в качестве модели драйверов устройств. Оно делает возможным портирование драйверов Linux (DDE/Linux) и FreeBSD (DDE/FreeBSD) в L4 userlevel. Таким образом, огромная кодовая база драйверов Linux может быть использована в будущем. В отличие от драйверов Windows, они доступны в виде исходного кода и могут быть портированы для работы в user-level.

## Зачем нужна реимплементация OS/2 с нуля?

OS/2 имеет одно из самых стабильных, надежных и высокопроизводительных ядер. Написанное примерно на 40% на ассемблере, оно очень хорошо оптимизировано и на 100% задействует возможности архитектуры i386. Его модульная структура позволяет легко заменять компоненты более улучшенными и менее ресурсоемкими, отрывать GUI, или настраивать систему под надобности пользователя. Она является очень гибко настраиваемой. Нам нравится ее компактное и “вылизанное” API, простота использования и интуитивный гибкий объектно-ориентированный интерфейс. Она использует один из лучших общеупотребительных скриптовых языков – REXX – в качестве встроенного в систему и многие приложения скриптового механизма. OS/2 рекламировалась фирмой IBM как “DOS лучше чем DOS и Windows лучше чем Windows”. Это правда – ее VDM была практически лучшей из существующих. И это верно не только для DOS/Windows. Поддержка Java и XFree86 была также очень мощной. Поэтому, мы полюбили OS/2 как мощную интегрирующую платформу (Integration Platform, (TM) by IBM) на основе единого десктопа. Она широко использовалась множеством маргиналов и нонконформистов многие годы и всегда имела свой собственный “way of doing things”. Мы хотим, продолжать следовать этому пути ;) Мы можем спать спокойно, зная, что наша система не популярна среди хакеров и вирусописателей – они обычно атакуют мейнстрим… Но мы не можем спокойно жить, как раньше – начиная с декабря 2006 года IBM-овское начальство решило похоронить OS/2, и оставило OS/2 сообществу быстро

устаревающую систему и ядро без исходников, и, самое главное, ухудшающаяся поддержка оборудования. Драйверов стали писать все меньше и меньше, и, в основном, новые драйвера являются портами с Linux. Петиции к IBM об открытии исходников также не дали результатов. Тем не менее, мы хотим продолжить жить и работать в OS/2. Самая важная задача – это написание нового ядра. Здесь следует отметить, что OS/2 до сих пор имеет 32-битное ядро. Существующее ядро, даже при наличии исходников, не переносимо на другие аппаратные платформы, главные из которых – ARM и x86_64. Современное ПО быстро разбухает, поэтому скоро мы должны столкнуться с ограничениями 32-битной архитектуры, самое главное из которых – невозможность использования более 4 Гб оперативной памяти. Уже сейчас веб-браузеры и офисы легко съедают гигабайты ОЗУ. Кроме того, одними из главных потребителей памяти являются виртуальные машины. А мы должны не забывать о позиционировании OS/2 как интеграционной платформы, в том числе и для виртуальных машин. Поэтому OS/2 требуется новое ядро. Нам всегда были интересны эксперименты IBM's с OS/2 поверх микроядра. Мы читали редбук от IBM про OS/2 Warp (PowerPC edition). Поэтому мы с энтузиазмом поддержали идею использования L4 как основы для реализации OS/2 API. Это дало толчок началу этого проекта.

## Почему бы не мигрировать на другую ОС?

IBM, Netlabs и другие компании пытаются заставить пользователей мигрировать на другую ОС – GNU/Linux, BSD, Windows и др. Мы согласны с тем, что TCO текущих версий OS/2 становится все больше и больше для домашнего пользователя. В области серверов OS/2 также быстро сдает позиции (но по-прежнему является зрелой и стабильной). Но нам нравится подход IBM к дизайну ОС. OS/2 нам нравится **продуманностью**, а не только ее дизайном, как он есть. Она проста в использовании. Ее API компактный и чистый. Мы хотим продолжать работать и программировать в OS/2.

## Совместимая

Планируется сделать osFree совместимой с основной частью текущего OS/2 API. Но мы не планируем поддерживать совместимость с API для драйверов. Драйверы OS/2 все больше и больше устаревают и не поддерживают множество новых устройств, но мы хотим иметь поддержку нового оборудования. Согласно своей идее, osFree может работать поверх множества ядер, таких как L4, Linux, Windows, etc. В результате, мы можем использовать подсистему драйверов от этого ядра.

Совместимость с OS/2 API позволит нам иметь то же самое чистое и компактное API и использовать существующие приложения. Мы пока не планируем полную поддержку всей 16-битной части OS/2, так как существует не так уж и много приложений, которые являются чисто 16-битными. Для большинства смешанных 16/32-бит приложений мы планируем сделать автоматическую замену “на лету” переходников (thunks) в 16 бит на вызовы чисто 32-битных API. Как результат, мы получим чисто 32-битные приложения (после мнголетней практики использования “смешанных” 16/32 приложений).

Конвертер приложений в чисто 32-битные можно встроить в загрузчик исполняемых файлов. Таким образом, в памяти всегда будет чисто 32-битное приложение, которое даже можно сохранить на диск и потом запускать из сохраненной копии (наподобие Just-in-time-компиляции в Java).

Мы также планируем ограниченную поддержку DOS и Win16 personalities (исторически, OS/2 славилась поддержкой DOS/Windows приложений). Но, по ходу дела, возможно создание и других personalities.

## Легковесная

OS/2 – одна из самых легких и компактных 32-битных ОС. osFree также должна быть легковесной, насколько это возможно. Мы не должны требовать гигабайты памяти как минимум для комфортной работы. Должна быть возможность нормально работать на достаточно слабом оборудовании, насколько возможно. В этом случае (как бонус), она также сможет быть востребованной в области embedded приложений.

## Открытая

Компоненты osFree распространяются на условиях открытых лицензий, таких как BSD или (L)GPL. Также, мы будем пытаться документировать все интерфейсы настолько, насколько возможно. Так что возможно улучшать все эти компоненты как это угодно пользователю.

## Объектно-ориентированная

osFree делает попытку улучшить дизайн рабочего стола OS/2 и других частей системы, используя принципы ООП. Мы используем SOM как базовую объектную модель, как предусмотрено дизайном IBM для рабочего стола OS/2 и других частей системы. Кроме того, мы выдвигаем идею CPI+, GPI+, PM+, т.е., планируем сделать OO обертки API различных частей системы (на основе SOM) (CPI [1], GPI [2] и PM, соответственно). Также возможна реализация доступа к более низкоуровневым частям системы в виде SOM объектов.

## Помощь проекту

Нам предстоит много сделать, так что мы приветствуем любую помощь проекту. Не только помощь в разработке и написании кода, но и написание документации, поддержка веб-страниц, поддержка дистрибутива ОС, и многое другое. См. Дорожную карту проекта для более подробной информации о целях и задачах проекта, а также основных этапах его развития.

Мы также ищем разработчиков, желающих оказать помощь проекту. Для новичков, у нас есть довольно много простых задач. Если вы – опытный разработчик, то у нас есть множество сложных задач, которые могут раскрыть ваш талант. См. страницу для разработчиков для подробностей о разработке проекта, также вы можете ознакомиться с лицензиями на компоненты osFree.

Наш IRC канал #osFree в сети EFnet и eCSnet.

1)
OS/2 kernel Control Program Interface
2)
Graphics Programming Interface

From:
http://www.osfree.org/doku/ - **osFree wiki**

Permanent link:
**http://www.osfree.org/doku/doku.php?id=ru:about**

Last update: **2020/02/11 02:46**