Unix

История Unix

UNIX появился в 1973 (начал разрабатываться в 1969) в Bell Labs. Первая целевая платформа — миникомпьютеры DEC (PDP-7).

В Unix были созданы такие технологии, как: язык C, оператор `pipe` (`|`) для взаимодействия между процессами, интерфейс сокетов BSD и многие другие.

UNIX изначально был условно открытой системой, достаточно удобной для портирования на другие архитектуры. Поэтому довольно быстро появились разные ветки (варианты) Unix'ов. Первой такой веткой (fork'ом) стал Берклевский дистрибутив (BSD) в 1977 году. В то же время, лицензия UNIX не давала возможности неограниченного изменения и модификации системы, с чем были связанны многие юридические конфликты. В конце концов, на данный момент сформировалось несколько закрытых коммерческих версий Unix'а, несколько открытых версий, а также ряд Unix-подобных систем, созданных с нуля (прежде всего, GNU/Linux).

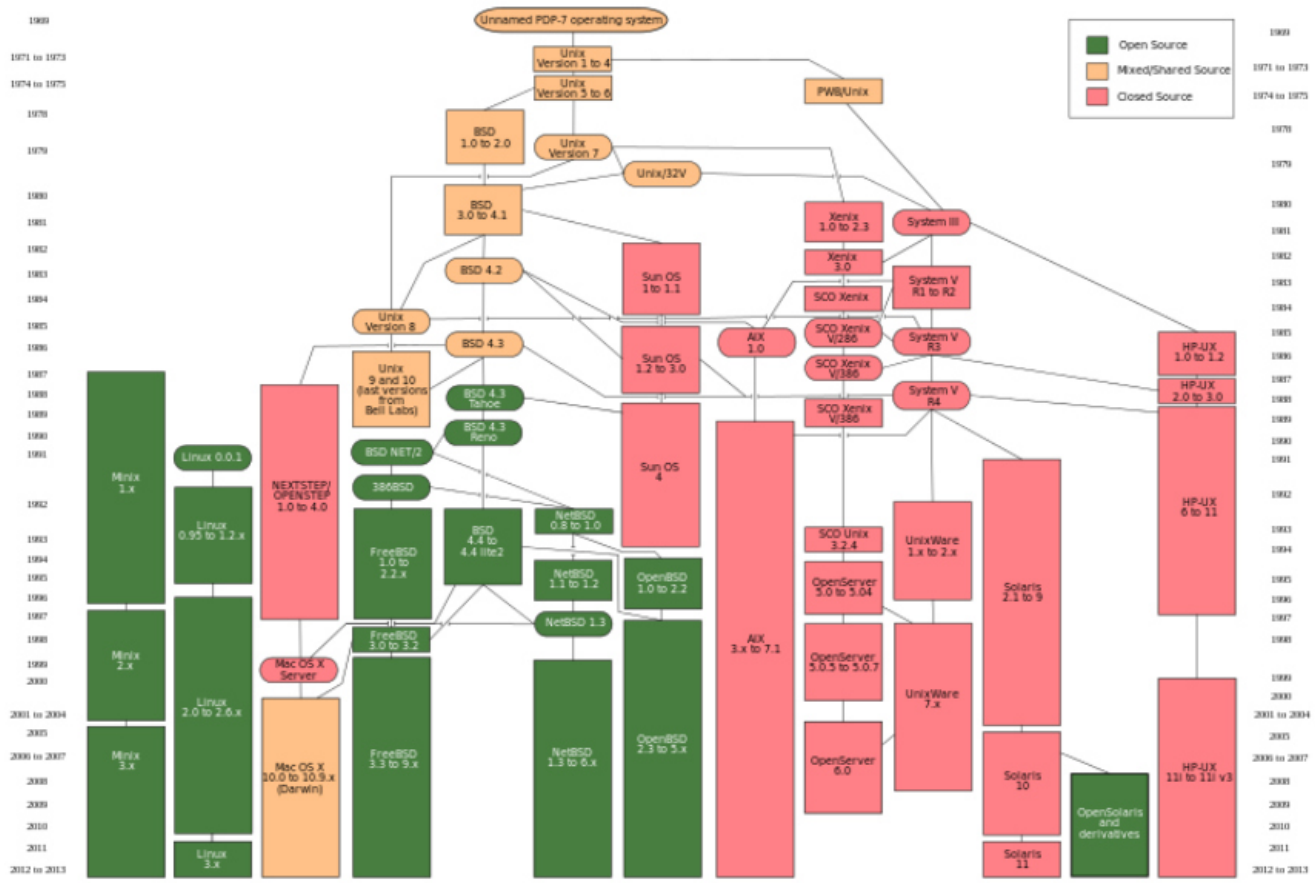

Рис. 11.1. Семейство потомков Unix и Unix-подобных ОС

* pdf generated on 2015-01-05 from markdown sources with SBCL 1.1.14 on Linux 3.11.0-12-generic with 3bmd, cl-pdf & cl-typesetting

Основные вариации Unix'ов

Коммерческие вариации

Коммерческие версии Unix ведут свою родословную от т.н. Unix System V. Практически все они прекратили свое развитие. Основные представители:

- SCO UnixWare (прекращен)
- Sun Solaris (прекращен)
- IBM AIX
- HP-UX

На данный момент развивается в качестве ОС для дата-центров только illumos — открытый наследник Sun Solars, который был самой продвинутой и активно изменяющейся версией Unix'a.

Ключевые технологии illumos:

- ФС ZFS
- технолония изоляции и создания т.н. песочниц Solaris Zones
- система интроспекции DTrace
- технология виртуализации Kernel Virtual Machine

Варианты BSD

Берклевский дистрибутив развивался как альтернатива System V Unix и с момента выхода в свет версии 4.3 Tahoe стал полностью открытым и бесплатным. С дистрибутивами BSD связанна одна из самых распространенных open-source лицензий — лицензия BSD. BSD-версии Unix продолжают активно развиваться.

Представители BSD семейства:

- FreeBSD
- OpenBSD
- NetBSD
- DragonflyBSD

На основе дистрибутива FreeBSD было создано ядро Darwin, которое используется в MacOS X.

* pdf generated on 2015-01-05 from markdown sources with SBCL 1.1.14 on Linux 3.11.0-12-generic with 3bmd, cl-pdf & cl-typesetting

GNU/Linux

Linux — это Unix и не Unix — полностью новая система, созданная на основе принципов Unix, как их понимал Линус Торвальдс. Толчком для написания системы стали распространение учебного варианта Unix'a MINIX, созданного Таненбаумом, а также появление процессора Intel 386 с поддержкой плоской сегментной модели, что радикально упростило написание ядра ОС.

Linux обошел другие варианты Unix'a на волне open-source за счет использования наработок движения GNU (компилятор gcc, утилиты core-utils, редактор Emacs и др.) и лицензии GPL, которая требует открытия доработок системы на тех же условиях, что и оригинальной системы.

Linux — это ядро, на основании которого создается дистрибутив — набор системных утилит и других программ, необходимых для работы системы, таких как графическая оболочка, офисные приложения и т.п. Существуют десятки дистрибутивов GNU/Linux, среди которых наиболее распространены ветки Red Hat (Red Hat Enterprise Linux, Fedora, Suse, CentOS) и Debian (Debian, Ubuntu, Mint).

Специфическим дистрибутивом Linux является ОС Google Android.

Plan 9

Plan 9 была академической попыткой в рамках лаборатории Bell Labs создать наследника Unix, в котором бы были исправлены его основные недостатки. Она оказалась классическим примером синдрома "второй системы", и не получила распространения.

Ключевые решения:

- всё – действительно файл
- отдельные пространства имен
- упразднение суперпользователя

Ключевые решения Unix

- открытая система (man, POSIX, open source)
- файл-центричность и текст-центричность
  - "Small pieces, loosely joined"
  - развитые средства IPC
- иерархическая файловая система с примитивной моделью безопасности

* pdf generated on 2015-01-05 from markdown sources with SBCL 1.1.14 on Linux 3.11.0-12-generic with 3bmd, cl-pdf & cl-typesetting

- плоское API системных вызовов, основанное на C
  - дополнительные возможности спрятаны в ioctl, fnctl, и т.п.
- пользователь root
- модель запуска процессов fork/exec
- развитая система управления зависимостями

Поддержка GUI в Unix

Исторически ядро Unix разрабатывось без поддержки графического интерфейса, который на тот момент еще не был развит. По мере развития различных вариантов графической аппаратной части в Unix было создано решения для поддержки и работы с ними — протокол X11, разработанный рабочей группой в университете MIT.

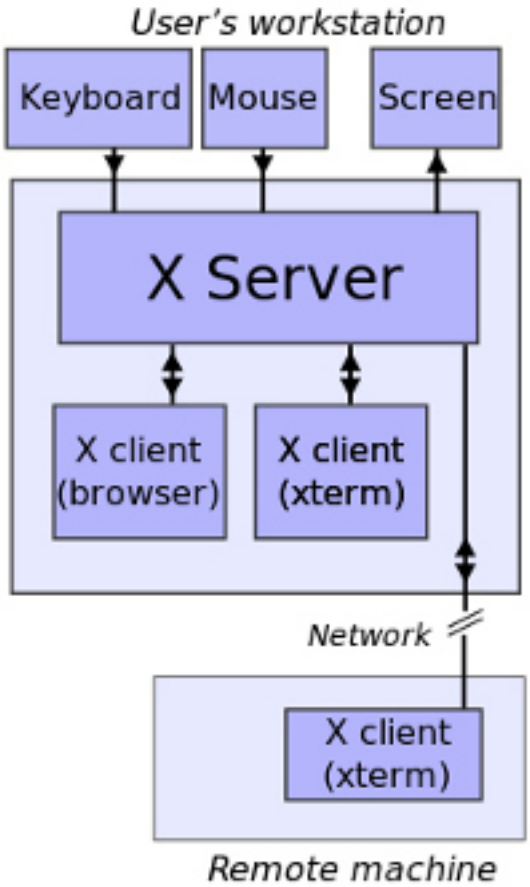

Рис. 11.2. Схема взаимодействия в рамках протокола X11

X11 протокол имеет различные реализации:

- xfree86
- X.org

* pdf generated on 2015-01-05 from markdown sources with SBCL 1.1.14 on Linux 3.11.0-12-generic with 3bmd, cl-pdf & cl-typesetting

- Wayland

На базе X11 протокола, как правило, строятся более высокоуровенвые графические фреймворки (GUI toolkits). Самые распространенные фреймворки в Unix-среде:

- GTK
- Qt

Оконные менеджеры решают задачу управления интерфейсом приложений в рамках единой концепции (на данный момент принята концепция Рабочего стола или WIMP). Как правило, оконные менеджеры строятся на основе графических фреймворков. Например, самый распространенный менеджер Gnome использует GTK, KDE построен на основе Qt.

Другие оконные менеджеры:

- обычные: Xfce, Sawfish
- плиточные: ion3, XMonad

Управление зависимостями

В условиях создания новых приложений с использованием созданной ранее базы програмных библиотек, важной и сложной задачей, которую решает любой дистрибутив Unix (а также, на данный момент и среда всех языков программирования), является управление зависимостями. Для этого (почти) каждый дистрибутив Unix имеет специальную программу — пактеный менеджер, который отвечает за ведение базы существующих библиотек и их версий, зависимостей и совместимости между версиями, а также мест хранения исходного или объектного кода, из которых можно загрузить ту или иную версию.

Различают менеджеры зависимостей, которые работают на уровне исходного кода (с его последующей сборкой) и на уровне бинарных файлов.

Самые распространенные пакетные менеджеры:

- менеджеры на основе формата APT в Debian
- менеджеры на основе формата RPM в Red Hat
- BSD ports и система Portage Gentoo Linux

Также, большинство сред языков программирования предоставляют свой особый менеджер пакетов. Например:

* pdf generated on 2015-01-05 from markdown sources with SBCL 1.1.14 on Linux 3.11.0-12-generic with 3bmd, cl-pdf & cl-typesetting

- Maven в Java
- RubyGems в Ruby
- NPM в Node.js

Принципы разработки под Unix

В книге "Искусство программирования под Unix" (TAOUP) Эрик Реймонд сформулировал следующие принципы хорошего тона, которые предпочтительно применяются при разработке как самих частей Unix систем, так и программ для Unix:

- Модульности (Modularity): создавать простые части, связываемые чистыми интерфейсами
- Ясности (Clarity): ясные решения лучше хитрых и умных
- Композиции (Composition): создавать программы, которые могут быть связанными с другими программами
- Разделения (Separation): разделять политику и механизм, интерфейс и реализацию
- Простоты (Simplicity): придумывать самое простое решение, добавлять сложность только по необходимости
- Бережливости (Parsimony): создавать большие программы, только когда продемонстрированно, что другие не справятся
- Прозрачности (Transparency): продумывать программы с тем, чтобы сделать интроспецкцию и отладку возможной и максимально простой
- Прочности (Robustness): прочность — это дитя прозрачности и простоты
- Представлений (Representation): вкладывать знания в данные, чтобы логика программы была тупой и надежной
- Наименьшего удивления (Least Surprise): при проектировании интерфейсов всегда нужно выбирать наименее неожиданный вариант
- Тишины (Silence): когда у программы нет ничего удивительного, чтобы сообщить, она не должна сообщать ничего
- Починки (Repair): стараться починить все, что можно, но когда это не удается — падать, падать громко и как можно раньше
- Экономии (Economy): время программиста дорого стоит, его лучше сберечь и потратить машинное время
- Генерации (Generation): по возможности избегать кодирования, когда можно написать программу, которая будет писать другие программы

* pdf generated on 2015-01-05 from markdown sources with SBCL 1.1.14 on Linux 3.11.0-12-generic with 3bmd, cl-pdf & cl-typesetting

- Оптимизации (Optimization): прототипировать перед шлифовкой, добиться работы программы перед ее оптимизацией
- Разнообразия (Diversity): не доверять никаким утверждениям о единственном верном пути
- Расширяемости (Extensibility): проектировать на будущее — оно наступит раньше, чем мы думаем

Критика Unix

Наряду с фанатами Unix существуют и Unix-ненависники, которые даже написали собственную книгу — Руководство Unix-ненависника.

Вот некоторые типичные претензии к Unix системам:

- Слабая поддержка GUI
- Слабая поддержка сценариев работы обычного пользователя (т.н. Desktop сценарии)
- Недостаточное следование модели “всё – файл”
- Примитивная модель безопасности: простой ACL, пользователь root
- Примитивная файловая система
- Дополнительные возможности спрятаны в ioctl, fnctl, ...
- Устаревший системный язык (C)

Литература

- The Art of Unix Programming
- The UNIX-HATERS Handbook
- The Daemon, the GNU & the Penguin

* pdf generated on 2015-01-05 from markdown sources with SBCL 1.1.14 on Linux 3.11.0-12-generic with 3bmd, cl-pdf & cl-typesetting