Оглавление

[1. Что такое ОС? Взгляды "снизу" (управление ресурсами), "сверху" (абстракции), "со стороны" (эксплуатация, жизненный цикл). 2](#_Toc472523021)

[2. История развития ОС. 4](#_Toc472523022)

[3. Отечественные ОС. 10](#_Toc472523023)

[4. Свободные лицензии (BSD, GPL разных версий и др.). Свободный код не означает общедоступный или бесплатный. Совместимость, возможный переток кода, примеры. 13](#_Toc472523024)

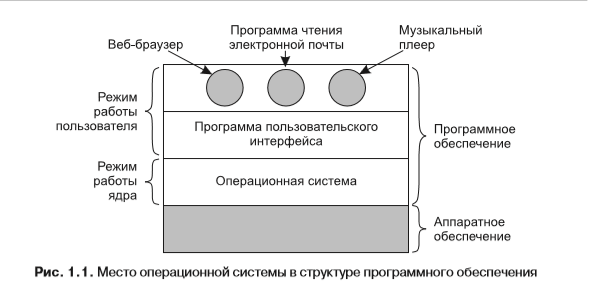
[5. Коммерческие лицензии (OEM, BOX, аренда, подписка). Примеры. 15](#_Toc472523025)

[6. Классификация ОС. Не менее пяти критериев, пояснить выбор и содержание критериев. 17](#_Toc472523026)

[7. Процессы (абстракция). Состояния (жизненный цикл) процессов, создание, завершение. Реализация (таблица процессов). Модель многозадачности, оценка утилизации ресурсов процессора, накладные расходы. Процесс как контейнер ресурсов. 20](#_Toc472523027)

1. Что такое ОС? Взгляды "снизу" (управление ресурсами), "сверху" (абстракции), "со стороны" (эксплуатация, жизненный цикл).

Компьютеры оснащены специальным уровнем программного обеспечения, который называется операционной системой, в чью задачу входит управление пользовательскими программами, а также всеми ранее упомянутыми ресурсами (одного или нескольких процессоров, оперативной памяти, дисков, принтера, клавиатуры, мыши, дисплея, сетевых интерфейсов и других разнообразных устройств ввода-вывода). Программы, с которыми взаимодействуют пользователи, обычно называемые оболочкой, когда они основаны на применении текста, и графическим пользовательским интерфейсом (Graphical User Interface (GUI)), когда в них используются значки, фактически не являются частью операционной системы, хотя задействуют эту систему в своей работе. Большинство компьютеров имеют два режима работы: режим ядра и режим пользователя. Операционная система — наиболее фундаментальная часть программного обеспечения, работающая в режиме ядра (этот режим называют еще режимом супервизора). В этом режиме она имеет полный доступ ко всему аппаратному обеспечению и может задействовать любую инструкцию, которую машина в состоянии выполнить. Вся остальная часть программного обеспечения работает в режиме пользователя, в котором доступно лишь подмножество инструкций машины.



Все программы, работающие в режиме ядра, безусловно, являются частью операционной системы, но некоторые программы, работающие вне этого режима, возможно, также являются ее частью или, по крайней мере, имеют с ней тесную связь.

Понятно, почему операционные системы живут так долго, — их очень трудно создавать, и, написав одну такую систему, владелец не испытывает желания ее выбросить и приступить к созданию новой. Поэтому операционные системы развиваются в течение долгого периода времени. Семейство Windows 95/98/Me по своей сути представляло одну операционную систему, а семейство Windows NT/2000/XP/Vista/ Windows 7 — другую.

Дать точное определение операционной системы довольно трудно. Можно сказать, что это программное обеспечение, которое работает в режиме ядра, но и это утверждение не всегда будет соответствовать истинному положению вещей. Отчасти проблема здесь в том, что операционные системы осуществляют две значительно отличающиеся друг от друга функции: предоставляют прикладным программистам (и прикладным программам, естественно) вполне понятный абстрактный набор ресурсов взамен неупорядоченного набора аппаратного обеспечения и управляют этими ресурсами.

Сверху:  
Абстракции: пример – жесткий диск->драйвера->файлы.

Хорошая абстракция превращает практически неподъемную задачу в две, решить которые вполне по силам. Первая из этих задач состоит в определении и реализации абстракций, а вторая — в использовании этих абстракций для решения текущей проблемы. Задача операционной системы заключается в создании хорошей абстракции, а затем в реализации абстрактных объектов, создаваемых в рамках этой абстракции, и управлении ими.

Снизу:  
Управление ресурсами (менеджер ресурсов).

Современные компьютеры состоят из процессоров, памяти, таймеров, дисков, мышей, сетевых интерфейсов, принтеров и широкого спектра других устройств. Сторонники взгляда снизу вверх считают, что задача операционной системы заключается в обеспечении упорядоченного и управляемого распределения процессоров, памяти и устройств ввода-вывода между различными программами, претендующими на их использование.

Современные операционные системы допускают одновременную работу нескольких программ. Сторонники этого взгляда на операционную систему считают, что ее первичной задачей является отслеживание того, какой программой какой ресурс используется, чтобы удовлетворять запросы на использование ресурсов, нести ответственность за их использование и принимать решения по конфликтующим запросам от различных программ и пользователей.

Управление ресурсами включает в себя мультиплексирование (распределение) ресурсов двумя различными способами: во времени и в пространстве. Когда ресурс разделяется во времени, различные программы или пользователи используют его по очереди: сначала ресурс получают в пользование одни, потом другие и т. д. К примеру, располагая лишь одним центральным процессором и несколькими программами, стремящимися на нем выполняться, операционная система сначала выделяет центральный процессор одной программе, затем, после того как она уже достаточно поработала, центральный процессор получает в свое распоряжение другая программа, затем еще одна программа, и, наконец, его опять получает в свое распоряжение первая программа. Определение того, как именно ресурс будет разделяться во времени — кто будет следующим потребителем и как долго, — это задача операционной системы. Другим примером мультиплексирования во времени может послужить совместное использование принтера. Когда в очереди для распечатки на одном принтере находятся несколько заданий на печать, нужно принять решение, какое из них будет выполнено следующим. Другим видом разделения ресурсов является пространственное разделение. Вместо поочередной работы каждый клиент получает какую-то часть разделяемого ресурса. Например, оперативная память обычно делится среди нескольких работающих программ, так что все они одновременно могут постоянно находиться в памяти (например, используя центральный процессор по очереди). При условии, что памяти достаточно для хранения более чем одной программы, эффективнее разместить в памяти сразу несколько программ, чем выделять всю память одной программе, особенно если ей нужна лишь небольшая часть от общего пространства. Разумеется, при этом возникают проблемы равной доступности, обеспечения безопасности и т. д., и их должна решать операционная система. Другим ресурсом с разделяемым пространством является жесткий диск. На многих системах на одном и том же диске могут одновременно храниться файлы, принадлежащие многим пользователям. Распределение дискового пространства и отслеживание того, кто какие дисковые блоки использует, — это типичная задача операционной системы по управлению ресурсами.

1. История развития ОС.

Так как операционные системы появились и развивались в процессе конструирования компьютеров, эти события исторически тесно связаны.

**Первое поколение (1945–1955): электронные лампы.**

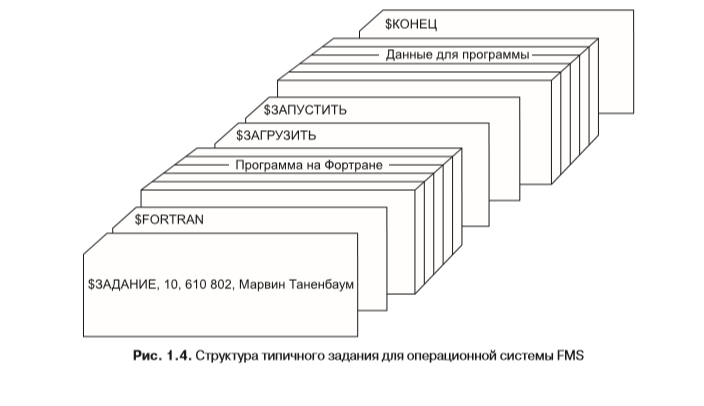
На заре компьютерной эры каждую машину проектировала, создавала, программировала, эксплуатировала и обслуживала одна и та же группа людей (как правило, инженеров). Все программирование велось исключительно на машинном языке или, и того хуже, за счет сборки электрических схем, а для управления основными функциями машины приходилось подключать к коммутационным панелям тысячи проводов. О языках программирования (даже об ассемблере) тогда еще ничего не было известно. Об операционных системах вообще никто ничего не слышал. Когда в начале 1950-х годов появились перфокарты, положение несколько улучшилось. Появилась возможность вместо использования коммутационных панелей записывать программы на картах и считывать с них, но в остальном процедура работы не претерпела изменений.

**Второе поколение (1955–1965): транзисторы и системы пакетной обработки.**

В середине 1950-х годов изобретение и применение транзисторов радикально изменило всю картину. Компьютеры стали достаточно надежными, появилась высокая вероятность того, что машины будут работать довольно долго, выполняя при этом полезные функции. Впервые сложилось четкое разделение между проектировщиками, сборщиками, операторами, программистами и обслуживающим персоналом.

Если учесть высокую стоимость оборудования, неудивительно, что люди довольно скоро занялись поиском способа повышения эффективности использования машинного времени. Общепринятым решением стала система пакетной обработки. Первоначально замысел состоял в том, чтобы собрать полный поднос заданий (колод перфокарт) в комнате входных данных и затем переписать их на магнитную ленту, используя небольшой и (относительно) недорогой компьютер, например IBM 1401, который был очень хорош для считывания карт, копирования лент и печати выходных данных, но не подходил для числовых вычислений.





Большие компьютеры второго поколения использовались главным образом для научных и технических вычислений, таких как решение дифференциальных уравнений в частных производных, часто встречающихся в физике и инженерных задачах. В основном программы для них составлялись на языке Фортран и ассемблере, а типичными операционными системами были FMS (Fortran Monitor System) и IBSYS (операционная система, созданная корпорацией IBM для компьютера IBM 7094).

**Третье поколение (1965–1980): интегральные схемы и многозадачность.**

К началу 1960-х годов большинство производителей компьютеров имели два различных, не совместимых друг с другом семейства. С одной стороны, это были огромные научные компьютеры с пословной обработкой данных типа IBM 7094, которые использовались для промышленного уровня числовых расчетов в науке и технике, с другой — коммерческие компьютеры с посимвольной обработкой данных, такие как IBM 1401, широко используемые банками и страховыми компаниями для задач сортировки и распечатки данных. Развитие и поддержка двух совершенно разных семейств была для производителей весьма обременительным делом.

Фирма IBM попыталась решить эти проблемы разом, выпустив серию машин IBM System/360. Семейство компьютеров IBM/360 стало первой основной серией, использующей малые интегральные схемы, дававшие преимущество в цене и качестве по сравнению с машинами второго поколения, собранными на отдельных транзисторах.

По замыслу его создателей, все программное обеспечение, включая операционную систему OS/360, должно было одинаково хорошо работать на всех моделях компьютеров. Но ни IBM, ни кому-либо еще так и не удалось создать программное обеспечение, удовлетворяющее всем этим противоречивым требованиям. В результате появилась громоздкая и чрезвычайно сложная операционная система, примерно на два или три порядка превышающая по объему FMS. Она состояла из миллионов строк, написанных на ассемблере тысячами программистов, содержала тысячи и тысячи ошибок, что повлекло за собой непрерывный поток новых версий, в которых предпринимались попытки исправления этих ошибок. В каждой новой версии устранялась только часть ошибок, вместо них появлялись новые, так что общее их количество, скорее всего, оставалось постоянным. Несмотря на свой огромный объем и имеющиеся недостатки, OS/360 и подобные ей операционные системы третьего поколения, созданные другими производителями компьютеров, неплохо отвечали запросам большинства клиентов. Они даже сделали популярными несколько ключевых технических приемов, отсутствовавших в операционных системах второго поколения. Самым важным достижением явилась многозадачность.

Решение заключалось в разбиении памяти на несколько частей, называемых разделами, в каждом из которых выполнялось отдельное задание (рис. 1.5). Пока одно задание ожидало завершения работы устройства ввода-вывода, другое могло использовать центральный процессор. Если в оперативной памяти содержалось достаточное количество заданий, центральный процессор мог быть загружен почти на все 100 % времени. Множество одновременно хранящихся в памяти заданий требовало наличия специального оборудования для защиты каждого задания от возможного незаконного присваивания областей памяти и нанесения вреда со стороны других заданий. Для этой цели компьютеры 360-й серии и другие системы третьего поколения были оборудованы специальными аппаратными средствами.

Другим важным плюсом операционных систем третьего поколения стала способность считывать задание с перфокарт на диск по мере того, как их приносили в машинный зал. При окончании выполнения каждого текущего задания операционная система могла загружать новое задание с диска в освободившийся раздел памяти и запускать это задание. Этот технический прием называется подкачкой данных, или спулингом.

Хотя операционные системы третьего поколения неплохо справлялись с большинством научных вычислений и крупных коммерческих задач по обработке данных, но по своей сути они были все еще разновидностью систем пакетной обработки. В системах третьего поколения промежуток времени между передачей задания и возвращением результатов часто составлял несколько часов, так что единственная поставленная не в том месте запятая могла стать причиной сбоя при компиляции, и получалось, что программист полдня тратил впустую.

Желание сократить время ожидания ответа привело к разработке режима разделения времени — варианту многозадачности, при котором у каждого пользователя есть свой диалоговый терминал. Первая универсальная система с режимом разделения времени CTSS (Compatible Time Sharing System) была разработана в Массачусетском технологическом институте (M.I.T.) на специально переделанном компьютере IBM 7094 (Corbato et al., 1962). Однако режим разделения времени не стал действительно популярным до тех пор, пока на машинах третьего поколения не получили широкого распространения необходимые технические средства защиты.

После создания успешной системы CTSS Массачусетский технологический институт, исследовательские лаборатории Bell Labs и корпорация General Electric (главный на то время изготовитель компьютеров) решили начать разработку универсальной общей компьютерной системы — машины, которая должна была поддерживать одновременную работу сотен пользователей в режиме разделения времени. За основу была взята система распределения электроэнергии. Когда вам нужна электроэнергия, вы просто вставляете штепсель в розетку и получаете столько энергии, сколько вам нужно. Проектировщики этой системы, известной как MULTICS (MULTiplexed Information and Computing Service — мультиплексная информационная и вычислительная служба), представляли себе одну огромную вычислительную машину, воспользоваться услугами которой мог любой проживающий в окрестностях Бостона человек.

К концу XX века идея создания такого общего компьютера уже выдохлась, но она может возродиться в виде облачных вычислений, когда относительно небольшие компьютеры (включая смартфоны, планшеты и им подобные устройства) подключены к серверам, принадлежащим огромным удаленным центрам обработки данных, где и производятся все вычисления, а локальный компьютер используется просто для обслуживания пользовательского интерфейса.

Несмотря на коммерческую неудачу, система MULTICS оказала существенное влияние на последующие операционные системы (особенно на UNIX и ее производные, на FreeBSD, Linux, IOS и Android).

Еще одной важной разработкой времен третьего поколения были мини-компьютеры, невероятный взлет популярности которых начался с выпуска корпорацией DEC машины PDP-1. Компьютеры PDP-1 обладали оперативной памятью, состоящей всего лишь из 4 К 18-битовых слов, но стоили всего 120 тыс. долларов за одну машину (это меньше 5 % от цены IBM 7094) и поэтому расхватывались как горячие пирожки. Некоторые виды нечисловой работы они выполняли так же быстро, как и машины IBM 7094, в результате чего родилась новая отрасль производства. За этой машиной вскоре последовала целая серия компьютеров PDP других моделей (в отличие от семейства IBM, полностью несовместимых), и как кульминация появилась модель PDP-11. Кен Томпсон (Ken Thompson), один из ведущих специалистов Bell Labs, работавший над проектом MULTICS, чуть позже нашел мини-компьютер PDP-7, которым никто не пользовался, и решил написать упрощенную однопользовательскую версию системы MULTICS. Эта работа позже переросла в операционную систему **UNIX®**, ставшую популярной в академических кругах, правительственных учреждениях и во многих компаниях.

Чтобы появилась возможность писать программы, работающие в любой UNIX-системе, Институт инженеров по электротехнике и электронике (IEEE) разработал стандарт системы UNIX, названный POSIX, который в настоящее время поддерживается большинством версий UNIX. Стандарт POSIX определяет минимальный интерфейс системных вызовов, который должны поддерживать совместимые с ним системы UNIX. Фактически на данный момент POSIX-интерфейс поддерживается также рядом других операционных систем.

Желание получить свободно распространяемую версию MINIX (в отличие от образовательной) привело к тому, что финский студент Линус Торвальдс (Linus Torvalds) создал систему Linux.

**Четвертое поколение (с 1980 года по наши дни): персональные компьютеры.**

Следующий период эволюции операционных систем связан с появлением БИС — больших интегральных схем (LSI, Large Scale Integration) — кремниевых микросхем, содержащих тысячи транзисторов на одном квадратном сантиметре. С точки зрения архитектуры персональные компьютеры (первоначально называемые микрокомпьютерами) были во многом похожи на мини-компьютеры класса PDP-11, но, конечно же, отличались по цене(дешевле).

В 1974 году, когда корпорация Intel выпустила Intel 8080 — первый универсальный 8-разрядный центральный процессор, — для него потребовалась операционная система, с помощью которой можно было бы протестировать новинку. Корпорация Intel привлекла к разработкам и написанию нужной операционной системы одного из своих консультантов Гэри Килдэлла. Килдэлл создал дисковую операционную систему, названную CP/M (Control Program for Microcomputers — управляющая программа для микрокомпьютеров). Когда Килдэлл заявил о своих правах на CP/M, корпорация Intel удовлетворила его просьбу, поскольку не думала, что у микрокомпьютеров с диском есть будущее. Позже Килдэлл создал компанию Digital Research для дальнейшего развития и продажи CP/M.

В начале 1980-х корпорация IBM разработала IBM PC (Personal Computer — персональный компьютер)1 и начала искать для него программное обеспечение. Сотрудники IBM связались с Биллом Гейтсом, чтобы получить лицензию на право использования его интерпретатора языка Бейсик. Корпорация IBM снова обратилась к Гейтсу с просьбой обеспечить ее операционной системой. После повторного обращения Гейтс выяснил, что у местного изготовителя компьютеров, Seattle Computer Products, есть подходящая операционная система DOS (Disk Operating System — дисковая операционная система). Он направился в эту компанию с предложением выкупить DOS (предположительно за $50 000), которое компания Seattle Computer Products с готовностью приняла. Затем Гейтс создал пакет программ DOS/BASIC, и пакет был куплен IBM. Когда корпорация IBM захотела внести в операционную систему ряд усовершенствований, Билл Гейтс пригласил для этой работы Тима Патерсона (Tim Paterson), человека, написавшего DOS и ставшего первым служащим Microsoft. Видоизмененная система была переименована в MS-DOS (MicroSoft Disk Operating System) и быстро заняла доминирующее положение на рынке IBM PC.

CP/M, MS-DOS и другие операционные системы для первых микрокомпьютеров полностью основывались на командах, вводимых пользователем с клавиатуры. Со временем благодаря исследованиям, проведенным в 1960-е годы Дагом Энгельбартом (Doug Engelbart) в научно-исследовательском институте Стэнфорда (Stanford Research Institute), ситуация изменилась. Энгельбарт изобрел графический интерфейс пользователя (GUI, Graphical User Interface) вкупе с окнами, значками, системами меню и мышью.

Однажды Стив Джобс увидел GUI и сразу понял уровень заложенного в него потенциала. Джобс приступил к созданию компьютера Apple, оснащенного графическим пользовательским интерфейсом. Этот проект привел к созданию компьютера Lisa, который оказался слишком дорогим и не имел коммерческого успеха. Вторая попытка Джобса, компьютер Apple Macintosh, имел огромный успех потому, что обладал более дружественным пользовательским интерфейсом, предназначенным для пользователей, не разбиравшихся в компьютерах. В 1999 создана **Mac OS X,** которая является операционной системой, построенной на основе UNIX, хотя и с весьма своеобразным интерфейсом.

Когда корпорация Microsoft решила создать преемника MS-DOS, она была под большим впечатлением от успеха Macintosh. В результате появилась основанная на применении графического интерфейса пользователя система под названием Windows, первоначально являвшаяся надстройкой над MS-DOS. На протяжении примерно 10 лет, с 1985 по 1995 год, Windows была просто графической оболочкой, работавшей поверх MS-DOS. Однако в 1995 году была выпущена самостоятельная версия Windows — Windows 95. Она непосредственно выполняла большинство функций операционной системы, используя входящую в ее состав систему MS-DOS только для загрузки, а также для выполнения старых программ, разработанных для MS-DOS. В 1998 году была выпущена слегка модифицированная версия этой системы, получившая название Windows 98. Тем не менее обе эти системы, и Windows 95 и Windows 98, все еще содержали изрядное количество кода, написанного на ассемблере для 16-разрядных процессоров Intel. Другой операционной системой Microsoft была Windows NT (NT означает New Technology — новая технология), которая на определенном уровне совместима с Windows 95. Однако она была написана заново и представляла собой полноценную 32-разрядную систему. Ведущим разработчиком Windows NT был Дэвид Катлер. Пятая версия Windows NT была в начале 1999 года переименована в Windows 2000. Она предназначалась для замены обеих версий — Windows 98 и Windows NT 4.0. Но полностью этим планам также не суждено было сбыться, поэтому Microsoft выпустила еще одну версию Windows 98 под названием Windows Me (Millennium edition — выпуск тысячелетия). В 2001 году была выпущена слегка обновленная версия Windows 2000, названная Windows XP. Эта версия выпускалась намного дольше, по существу заменяя все предыдущие версии Windows. После Windows 2000 Microsoft разбила семейство Windows на клиентскую и серверную линейки. Клиентская линейка базировалась на версии XP и ее последователях, а серверная включала Windows Server 2003 и Windows 2008. Чуть позже появилась и третья линейка, предназначенная для мира встроенных операционных систем.

Затем в январе 2007 года Microsoft выпустила окончательную версию преемника Windows XP под названием Vista. У нее был новый графический интерфейс, усовершенствованная система безопасности и множество новых или обновленных пользовательских программ. Microsoft надеялась, что она полностью заменит Windows XP, но этого так и не произошло. Вместо этого было получено большое количество критических отзывов и статей в прессе, главным образом из-за высоких системных требований, ограничительных условий лицензирования и поддержки технических средств защиты авторских прав (технологии, затрудняющей пользователям копирование защищенных материалов). С появлением Windows 7, новой и менее требовательной к ресурсам операционной системы, многие решили вообще пропустить Vista. В Windows 7 не было представлено слишком много новых свойств, но она была относительно небольшой по объему и довольно стабильной. Менее чем за три недели Windows 7 получила большую долю рынка, чем Vista за семь месяцев. В 2012 году Microsoft выпустила ее преемника — Windows 8, операционную систему с совершенно новым внешним видом, предназначенным для сенсорных экранов.

Другим основным конкурентом в мире персональных компьютеров является операционная система UNIX (и различные производные от этой системы). UNIX имеет более сильные позиции на сетевых и промышленных серверах, также она находит все более широкое распространение и на настольных компьютерах, ноутбуках, планшетных компьютерах и смартфонах. На компьютерах с процессором Pentium популярной альтернативой Windows для студентов и постоянно растущего числа корпоративных пользователей становится операционная система Linux.

Операционная система FreeBSD также является популярной производной от UNIX, порожденной проектом BSD в Беркли. Все современные компьютеры Macintosh работают на модифицированной версии FreeBSD (OS X). UNIX также является стандартом на рабочих станциях, оснащенных высокопроизводительными RISC-процессорами. Ее производные нашли широкое применение на мобильных устройствах, которые работают под управлением iOS 7 или Android.

В середине 1980-х годов начало развиваться интересное явление — рост сетей персональных компьютеров, работающих под управлением сетевых операционных систем и распределенных операционных систем (Tanenbaum and Van Steen, 2007). В сетевых операционных системах пользователи знают о существовании множества компьютеров и могут войти в систему удаленной машины и скопировать файлы с одной машины на другую. На каждой машине работает своя локальная операционная система и имеется собственный локальный пользователь (или пользователи).

**Пятое поколение (с 1990 года по наши дни): мобильные компьютеры**

Первый настоящий мобильный телефон появился в 1946 году, и тогда он весил около 40 кг.

Первый по-настоящему переносной телефон появился в 1970-х годах и при весе приблизительно 1 кг был воспринят весьма позитивно. Первый настоящий смартфон появился только в середине 1990-х годов, когда Nokia выпустила свой N9000, представлявший собой комбинацию из двух отдельных устройств: телефона и КПК. В 1997 году в компании Ericsson для ее изделия GS88 «Penelope» был придуман термин «смартфон».

На момент написания этих строк доминирующей была операционная система Google Android, а на втором месте находилась Apple iOS.

В первое десятилетие после своего появления большинство смартфонов работало под управлением Symbian OS. Эту операционную систему выбрали такие популярные бренды, как Samsung, Sony Ericsson, Motorola и Nokia.

Для производителей телефонов Android обладала тем преимуществом, что имела открытый исходный код и была доступна по разрешительной лицензии. Кроме того, у этой операционной системы имеется огромное сообщество разработчиков, создающих приложения в основном на общеизвестном языке программирования Java.

1. Отечественные ОС.

При анализе истории развития области ИТ следует иметь в виду особые условия, в которых развивались эти разработки как в СССР (России), так и в США, начиная с 1950-х гг. – "холодная война" и "железный занавес". Вследствие этого, все эти разработки, как в области аппаратуры, так и в области программного обеспечения, были строго засекречены. Такая ситуация приводила к тому, что аналогичные идеи подчас возникали и реализовывались по обе стороны "железного занавеса" примерно в одно и то же время, при почти полном отсутствии информации о работах друг друга. Отечественные разработчики, почти ничего не зная об аналогичных работах американских коллег, создавали свои оригинальные системы, в том числе – ОС. Например, идея многопоточности (multi-threading) была реализована в ОС "Эльбрус" еще в конце 1970-х гг., а в популярных зарубежных ОС (UNIX, Solaris, Windows NT) многопоточность появилась только в конце 1980-х – начале 1990-х гг. К сожалению, имело место и существенное отставание советских и российских ИТ-специалистов от американцев – прежде всего, в области разработки элементной базы и технологии производства компьютеров, а также в области графических пользовательских интерфейсов (GUI).

Среди передовых оригинальных отечественных разработок в области компьютерной аппаратуры и ОС 1960-х – 1970-х гг. следует выделить прежде всего ЭВМ БЭСМ-6, ее операционные системы: ОС ДИСПАК, ОС ДИАПАК, ОС ИПМ и ее системное и прикладное программное обеспечение. Разработчиком БЭСМ-6, ОС ДИСПАК и ОС ДИАПАК был Институт точной механики и вычислительной техники АН СССР под руководством академика Сергея Алексеевича Лебедева, основателя всей нашей отечественной вычислительной техники. Разработчик ОС ИПМ – Институт прикладной математики АН СССР. ЭВМ БЭСМ-6 и ее программное обеспечение следует признать уникальными. В их развитии участвовали многие академические и университетские коллективы не только СССР, но и зарубежных стран - достаточно вспомнить такие системы, как АЛГОЛ-ГДР - реализацию расширения Алгола-60 с развитыми математическими библиотеками, выполненную нашими коллегами из Германии, а также реализацию Паскаля для БЭСМ-6, разработанную специалистами из Польской Академии наук. Операционные системы для БЭСМ-6 поддерживали пакетный (с учетом приоритетов и ресурсов заданий) и диалоговый режимы взаимодействия с компьютером, страничную организацию виртуальной памяти, работу с внешними устройствами и телекоммуникационными каналами, работу в локальных сетях. К каждой БЭСМ-6 были подключены десятки терминалов, работавших под управлением диалоговых систем ДИМОН, ДЖИН и др. (это при объеме оперативной памяти БЭСМ-6 всего в 32 страницы по 4096 байтов и быстродействии до 1 млн. операций в секунду). Работу БЭСМ-6 и ее ОС отличала высокая надежность. Руководитель разработки ОС ДИСПАК – В.Ф. Тюрин.

Другой передовой отечественной разработкой 1970-х – 1980-х гг. была разработка многопроцессорных вычислительных комплексов (МВК) "Эльбрус-1" и "Эльбрус-2". В разработке программного обеспечения системы "Эльбрус" автор курса со своей командой принимал активное участие в течение 15 лет. Идейным вдохновителем проекта "Эльбрус" стал сам С.А. Лебедев, затем им руководили академик Всеволод Сергеевич Бурцев, а после него – чл.-корр. АН СССР Борис Арташесович Бабаян. Следует признать, что у "Эльбруса" были зарубежные прототипы и задолго до его появления были написаны академические зарубежные работы, заложившие научные основы подобных компьютерных архитектур. Однако разработчикам "Эльбруса" и его операционной системы удалось предложить и реализовать целый ряд собственных оригинальных идей и методов. Основными принципами "Эльбруса", как и его предшественников, являлись: теговая архитектура (каждое слово памяти, кроме данных, содержало тег – код типа данных, хранящихся в этом слове, по которому аппаратура контролировала правильность выполнения операции), динамизм и аппаратная поддержка типичных (подчас весьма сложных) последовательностей действий, используемых при реализации языков высокого уровня - например, вход в процедуру по указателю на нее, с установкой дисплей-регистров, ссылающихся на доступные процедуре области локальных данных. ОС "Эльбрус" поддерживала создание процессов и операции над ними, аналогичные тем, которые впоследствии в зарубежных разработках были названы многопоточностью (multi-threading); была реализована математическая (виртуальная) память с поддержкой страничного распределения виртуальной памяти (на диске) и сегментного распределения физической (оперативной) памяти. Динамизм выражался в том, что отсутствовала статическая линковка; все программы и модули загружались в память только динамически, при первом вызове. Также динамически, при первом запросе, по прерыванию, выделялся каждый массив математической памяти. Подобные принципы были для своего времени передовыми, использование тегов значительно повысило надежность. Однако, с современной точки зрения, идеологию "Эльбруса", по-видимому, нельзя считать гибкой и эффективной, так как все аппаратные операции и соответствующие действия ОС были реализованы в общем виде, и практически отсутствовала какая-либо возможность оптимизаций, например, для более быстрого вызова процедуры в случае отсутствия необходимости обращения к ее аргументам, для быстрого доступа к статической области памяти и т.д.

Были и другие интересные отечественные разработки новых архитектур компьютеров и их операционных систем, прежде всего - оригинальные специализированные компьютеры для различных применений и их операционные системы (в основном, по своему классу и назначению, они были системами реального времени).

В начале 1970-х годов в развитии отечественной вычислительной техники и ее системного программного обеспечения начался новый этап. Правительство СССР приняло беспрецедентное решение о создании, в качестве основной на достаточно долгий период времени отечественной серии - Единой Системы ЭВМ (ЕС ЭВМ) - путем копирования американских компьютеров серии IBM 360. Соответственно, все базовое системное программное обеспечение, в том числе и ОС, также было адаптировано к использованию в СССР (либо использовалось в оригинальном виде – с сообщениями на английском языке и т.д.). Это решение вызвало большие проблемы с финансированием у разработчиков отечественных архитектур компьютеров. Это также вызвало большие сложности у пользователей и разработчиков программного обеспечения, так как далеко не все хорошо владели английским языком. Появились, например, системы-обертки, обеспечивающие русскоязычный интерфейс: с их помощью все задания для ЕС писались с использованием русскоязычной мнемоники, затем конвертировались в англоязычный Job Control Language (язык управления заданиями IBM 360), а все сообщения, выдаваемые в качестве результатов, переводились на русский язык. Это было интересным подходом, однако не прижилось. Документация по IBM 360 постепенно была переведена на русский язык, появилась русскоязычная справочная и учебная литература по ЕС ЭВМ. К сожалению, отечественные аналоги аппаратуры IBM 360 – машины серии ЕС ЭВМ – оказались гораздо менее надежными, чем их прототипы. В течение еще нескольких лет было принято еще одно правительственное решение – об аналогичном копировании американских миникомпьютеров серий PDP 10 и PDP 11, под общим названием "Система Мини-ЭВМ" (СМ ЭВМ). Были выпущены компьютеры этой серии СМ-1, СМ-2, СМ-3 и СМ-4. Были и другие аналогичные работы по копированию зарубежных архитектур компьютеров и выпуске на этой основе отечественных аналогов. Копирование машин IBM 360 и PDP, с одной стороны, дало возможность советским программистам освоить новые развитые операционные системы, языки программирования, библиотеки программ, с другой – отбросило нашу отчественную вычислительную технику еще дальше назад.

Разумеется, история отечественных ОС на этом не закончилась. Например, в настоящее время ведется разработка отечественной свободно распространяемой операционной системы на базе Linux. Среди отечественных программистов многие являются специалистами весьма высокого уровня по операционным системам.

Фантом ОС — операционная система, разрабатываемая российской компанией Digital Zone с 2010 года. Система базируется на концепции персистентной виртуальной памяти, ориентирована на управляемый код и нацелена на применение в носимых и встроенных компьютерах. ОС Фантом — одна из немногих ОС, не опирающихся на классические концепции Unix-подобных систем (в отличие от их концепции «Всё есть файл», Фантом базируется на принципе «Всё есть объект»). Предполагается, что модель ОС Фантом позволяет и самой системе, и приложениям быть более простыми и, в то же время, более эффективными.

1. Свободные лицензии (BSD, GPL разных версий и др.). Свободный код не означает общедоступный или бесплатный. Совместимость, возможный переток кода, примеры.

Свободная лицензия (англ. free license) — такой лицензионный договор (ранее — «авторский договор»), условия которого содержат разрешения пользователю от правообладателя на конкретный перечень способов использования его произведения, которые дают ему четыре важнейшие свободы

Свободное ПО (free software) — это ПО, распространяемое на условиях свободной лицензии (либо находящееся в общественном достоянии), а свободная лицензия — это та, которая даёт пользователю право:

* Использовать программу
* Изменять её исходный код
* Распространять программу
* Распространять изменённые версии

Свободное и открытое ПО — это совсем не то же самое, что бесплатное. Свободное и открытое ПО — это не то же самое, что некоммерческое. Практически все сколько-нибудь крупные свободные проекты (Linux, gcc, glibc, Gnome, FreeBSD, clang и многие другие) являются коммерческими.

Двойное лицензирование — это (если не указано противного) две лицензии, соединённые союзом «или» (то же самое относится к тройному лицензированию и т. д.). Т. е. пользователь может выбрать, на условиях какой из этих лицензий он будет пользоваться материалом.

**Лицензия BSD** (англ. BSD license, Berkeley Software Distribution license — Программная лицензия университета Беркли) — это лицензионное соглашение, впервые применённое для распространения UNIX-подобных операционных систем BSD.

Существуют две основные версии лицензии BSD, которые необходимо различать: «оригинальная» и так называемая «модифицированная» (вторую в англоязычной литературе часто называют New BSD License).

Эти лицензии были подвергнуты ряду изменений, породив множество лицензий, обобщённо именуемых «лицензии типа BSD». «В настоящее время лицензии типа BSD являются одними из самых популярных лицензий для свободного программного обеспечения и используются для многих программ (помимо BSD-версий UNIX, для которых лицензия BSD была изначально создана).»

Лицензия BSD допускает проприетарное коммерческое использование ПО. Для ПО, выпущенного под этой лицензией, допускается встраивание в проприетарные коммерческие продукты. Работы, основанные на таком ПО, даже могут распространяться под проприетарными лицензиями (но всё же обязаны соответствовать требованиям лицензии). Наиболее заметные примеры таких программ — использование сетевого кода BSD в продуктах корпорации Microsoft, а также использование многих компонентов FreeBSD в операционной системе Mac OS X. Плюс недавнее использование ядра FreeBSD для создания PlayStation 4.

Можно применять к распространяемому продукту одновременно лицензию BSD и какую-то другую. Например, так было с самыми ранними версиями самой BSD, которая включала проприетарные материалы из AT&T.

Главная особенность BSD заключается в том, что она не только позволяет использовать продукт в сторонних разработках, но в отличие от GPL разрешает в дальнейшем сделать его закрытым.

**GNU General Public License**

Сокращенное наименование — GNU GPL или просто GPL. Согласно исследованию, проведенному компанией Black Duck Software, 70% всех открытых проектов распространяются под лицензией семейства GPL.

Эта лицензия предоставляет пользователям следующее:

* — право свободно запускать программу с любой целью;
* — право свободно изучать исходный текст программы;
* — право свободно модифицировать и улучшать программу;
* — право свободно распространять её копии.

При этом важно, что ни один пользователь, получивший подобные права, не должен каким-то образом ограничивать аналогичные права других пользователей. То есть если кто-то решил модифицировать программу, то распространять измененный вариант он должен на тех же условиях, на которых получил прототип. Произведение, получив свободу один раз, не может быть лишено ее впоследствии.

GNU Lesser GPL v2.1: Данная версия лицензии датируется 1999 годом и содержит одно огромное отличие от обычной лицензии GNU GPL: предназначенная для библиотек, лицензия позволяет использовать их в проприетарном программном обеспечении. Например, библиотеки GNU C распространяются под лицензией GNU Lesser GPL v2.1, для того, чтобы сторонние разработчики могли использовать их в своем ПО, свободном или коммерческом.

1. Коммерческие лицензии (OEM, BOX, аренда, подписка). Примеры.



Коробочная версия включает в себя лицензионное соглашение, сертификат подлинности, дистрибутив с программным продуктом, а для более ранних версий продуктов — также регистрационную карточку и документацию в печатном виде.[4] Подтверждением лицензионных прав пользователя является Сертификат подлинности (СОА), наклеенный на коробку.

В случае OEM-версии (Original Equipment Manufacturer — англ. производитель нового оборудования) покупатель приобретает оборудование с предустановленными программным обеспечением.[5] Приобретаемое программное обеспечение жёстко привязано к оборудованию и может эксплуатироваться только на нём. OEM-версии программного обеспечения распространяются только среди поставщиков компьютерного оборудования и могут быть проданы только ими и только вместе с оборудованием. Необходимым подтверждением лицензионных прав пользователя является сертификат подлинности, наклеенный на корпус ПК.

Ограничения OEM-лицензий

* - OEM-версия может использоваться только на том ПК, на котором она была первоначально установлена. Это главное ограничение, отличающее OEM-версии от версий, закупленных по другим способам лицензирования.
* - Если клиент приобретает новый компьютер, он должен приобрести для него новую копию OEM-продукта. Перенести на этот компьютер OEM-лицензию со старого ПК нельзя, даже если старый компьютер более не используется.
* - Если пользователь хочет передать или продать старый компьютер с установленной на нем OEM-версией ПО другому пользователю, то вместе с компьютером передается и все части OEM- лицензии.

Немного от майкрософт

Open Value Subscription – это идеально подходящая для малых и средних предприятий программа лицензирования, которая позволяет взять в аренду лицензии и каждый год вносить платеж только за те программы, которые нужны компании в данный момент. Вы можете как увеличивать количество лицензий, так и отказываться от тех, которые стали не нужны бизнесу. Главное преимущество и отличие от других программ лицензирования – это возможность уменьшать нужное количество лицензий, что идеально подойдет для предприятий, которые не могут на данном этапе строить долгосрочные планы касательно нужного ПО.

Особенности лицензирования OVS

При оформлении соглашения все компьютеры компании стандартизируются, т.е. оплачиваются лицензии на базовые продукты (хотя бы на одну категорию – ОС, клиентские лицензии или приложения) для всех ПК предприятия. Количество необходимых базовых лицензий регулируется каждый год, и зависит от количества ПК. Т.е. если компания приобрела дополнительные ПК, в соглашение добавляются лицензии, если ПК стало меньше – количество лицензий уменьшается (минимальное количество устройств на компанию должно быть не меньше 5). Подписка на дополнительные, не базовые продукты, оформляется в любое время, а оплачивается в месяц заказа.

А это отдельно, но похоже, мы имеем дело с одним и тем же

Подписка на лицензирование программного обеспечения предусматривает внесение ежемесячных или ежегодных платежей. Эта схема удобна компаниям, которые покупают более 10 лицензий. Она позволяет пользователям за минимальные начальные затраты получить практически все основные преимущества использования данного продукта.

1. Классификация ОС. Не менее пяти критериев, пояснить выбор и содержание критериев.

**(из наших лекций)**

ОС принято классифицировать по следующим критериям:

1. Однопользовательские/многопользовательские
2. Однозадачные/многозадачные  
   допольнительно многозадачные классифицируются следующим образом:

* вытесняющие
* невытесняюшие
* реального времени

1. сертифицированная/несесртифицированная
2. поддерживает виртуальную память/не поддерживает
3. кроссплатформенные/нет

Главным отличием многопользовательских систем от однопользовательских является наличие средств защиты информации каждого пользователя от несанкционированного доступа других пользователей.  
Следует заметить, что не всякая многозадачная система является многопользовательской, и не всякая однопользовательская ОС является однозадачной.

Однозадачные операционные системы в основном выполняют функцию предоставления пользователю виртуальной машины, делая более простым и удобным процесс взаимодействия пользователя с компьютером. Однозадачные ОС включают средства управления периферийными устройствами, средства управления файлами, средства общения с пользователем.  
Многозадачные операционные системы, кроме вышеперечисленных функций, управляют разделением совместно используемых ресурсов, таких как процессор, оперативная память, файлы и внешние устройства.

Основным различием вытесняющего и не вытесняющего вариантов многозадачности является степень централизации механизма планирования процессов. В случае не вытесняющей многозадачности механизм планирования процессов целиком сосредоточен в операционной системе, а в случае вытесняющей многозадачности он распределен между системой и прикладными программами.  
При не вытесняющей многозадачности активный процесс выполняется до тех пор, пока он сам, по собственной инициативе, не отдаст управление операционной системе для того, чтобы та выбрала из очереди другой готовый к выполнению процесс.  
При вытесняющей многозадачности решение о переключении процессора с одного процесса на другой принимается операционной системой, а не самим активным процессом.

**Или с Интуита**

Все многообразие существующих (и ныне не использующихся) ОС можно классифицировать по множеству различных признаков. Остановимся на основных классификационных признаках.

**1.По назначению** ОС делятся на универсальные и специализированные.   
Специализированные ОС, как правило, работают с фиксированным набором программ (функциональных задач). Применение таких систем обусловлено невозможностью использования универсальной ОС по соображениям эффективности, надежности, защищенности и т.п., а также вследствие специфики решаемых задач.   
Универсальные ОС рассчитаны на решение любых задач пользователей, но, как правило, форма эксплуатации вычислительной системы может предъявлять особые требования к ОС, т.е. к элементам ее специализации.

**2.По способу загрузки** можно выделить загружаемые ОС (большинство) и системы, постоянно находящиеся в памяти вычислительной системы. Последние, как правило, специализированные и используются для управления работой специализированных устройств (например, в БЦВМ баллистической ракеты или спутника, научных приборах, автоматических устройствах различного назначения и др.).

**3.По особенностям алгоритмов управления ресурсами**. Главным ресурсом системы является процессор, поэтому дадим классификацию по алгоритмам управления процессором, хотя можно, конечно, классифицировать ОС по алгоритмам управления памятью, устройствами ввода-вывода и.т.д.

**Поддержка многозадачности** (многопрограммности). По числу одновременно выполняемых задач ОС делятся на 2 класса: однопрограммные (однозадачные) – например, MS-DOS, MSX, и многопрограммные (многозадачные) – например, ОС ЕС ЭВМ, OS/360, OS/2, UNIX, Windows разных версий.  
Однопрограммные ОС предоставляют пользователю виртуальную машину, делая более простым и удобным процесс взаимодействия пользователя с компьютером. Они также имеют средства управления файлами, периферийными устройствами и средства общения с пользователем. Многозадачные ОС, кроме того, управляют разделением совместно используемых ресурсов (процессор, память, файлы и т.д.), это позволяет значительно повысить эффективность вычислительной системы.

**Поддержка многопользовательского режима**. По числу одновременно работающих пользователей ОС делятся: на однопользовательские (MS-DOS, Windows 3х, ранние версии OS/2) и многопользовательские (UNIX, Windows NT/2000/2003/XP/Vista).  
Главное отличие многопользовательских систем от однопользовательских – наличие средств защиты информации каждого пользователя от несанкционированного доступа других пользователей. Следует заметить, что может быть однопользовательская мультипрограммная система.

**Виды многопрограммной работы**. Специфику ОС во многом определяет способ распределения времени между несколькими одновременно существующими в системе процессами (или потоками). По этому признаку можно выделить 2 группы алгоритмов: не вытесняющая многопрограммность (Windows3.x, NetWare) и вытесняющая многопрограммность (Windows 2000/2003/XP, OS/2, Unix).  
В первом случае активный процесс выполняется до тех пор, пока он сам не отдает управление операционной системе. Во втором случае решение о переключении процессов принимает операционная система. Возможен и такой режим многопрограммности, когда ОС разделяет процессорное время между отдельными ветвями (потоками, волокнами) одного процесса.

**Многопроцессорная обработка**. Важное свойство ОС – отсутствие или наличие средств поддержки многопроцессорной обработки. По этому признаку можно выделить ОС без поддержки мультипроцессирования (Windows 3.x, Windows 95) и с поддержкой мультипроцессирования (Solaris, OS/2, UNIX, Windows NT/2000/2003/XP).  
Многопроцессорные ОС классифицируются по способу организации вычислительного процесса на асимметричные ОС (выполняются на одном процессоре, распределяя прикладные задачи по остальным процессорам) и симметричные ОС (децентрализованная система).

**4.По области использования и форме эксплуатации**. Обычно здесь выделяют три типа в соответствии с использованными при их разработке критериями эффективности:

* системы пакетной обработки (OS/360, OC EC);
* системы разделения времени (UNIX, VMS);
* системы реального времени (QNX, RT/11).

Первые предназначались для решения задач в основном вычислительного характера, не требующих быстрого получения результатов. Критерий создания таких ОС – максимальная пропуская способность при хорошей загрузке всех ресурсов компьютера. В таких системах пользователь отстранен от компьютера.  
Системы разделения времени обеспечивают удобство и эффективность работы пользователя, который имеет терминал и может вести диалог со своей программой.  
Системы реального времени предназначены для управления техническими объектами (станок, спутник, технологический процесс, например доменный и т.п.), где существует предельное время на выполнение программ, управляющих объектом.

**5.По аппаратной платформе** (типу вычислительной техники), для которой они предназначаются, операционные системы делят на следующие группы.

**Операционные системы для смарт-карт**. Некоторые из них могут управлять только одной операцией, например, электронным платежом. Некоторые смарт-карты являются JAVA-ориентированным и содержат интерпретатор виртуальной машины JAVA. Апплеты JAVA загружаются на карту и выполняются JVM-интерпретатором. Некоторые из таких карт могут одновременно управлять несколькими апплетами JAVA, что приводит к многозадачности и необходимости планирования.

**Встроенные операционные системы**. Управляют карманными компьютерами (lialm OS, Windows CE – Consumer Electronics – бытовая техника), мобильными телефонами, телевизорами, микроволновыми печами и т.п.

**Операционные системы для персональных компьютеров**, например, Windows 9.x, Windows ХР, Linux, Mac OSX и др.

**Операционные системы мини-ЭВМ**, например, RT-11 для PDP-11 – OC реального времени, RSX-11 M для PDP-11 – ОС разделения времени, UNIX для PDP-7.

**Операционные системы мэйнфреймов** (больших машин), например, OS/390, происходящая от OS/360 (IBM). Обычно ОС мэйнфреймов предполагает одновременно три вида обслуживания: пакетную обработку, обработку транзакций (например, работа с БД, бронирование авиабилетов, процесс работы в банках) и разделение времени.

**Серверные операционные системы**, например, UNIX, Windows 2000, Linux. Область применения – ЛВС, региональные сети, Intranet, Internet.

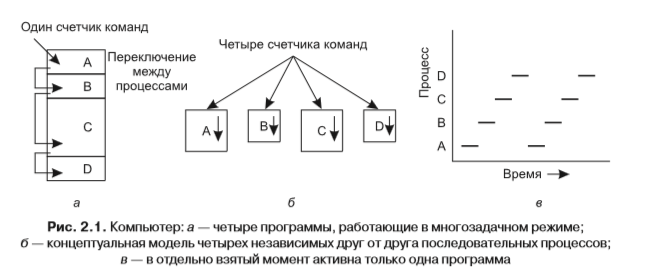
**Кластерные операционные системы**. Кластер – слабо связанная совокупность нескольких вычислительных систем, работающих совместно для выполнения общих приложений и представляющихся пользователю единой системной, например, Windows 2000 Cluster Server, Windows 2008 Server, Sun Cluster (базовая ОС – Solaris).

1. Процессы (абстракция). Состояния (жизненный цикл) процессов, создание, завершение. Реализация (таблица процессов). Модель многозадачности, оценка утилизации ресурсов процессора, накладные расходы. Процесс как контейнер ресурсов.

Основным понятием в любой операционной системе является процесс: абстракция, описывающая выполняющуюся программу. Процессы — это одна из самых старых и наиболее важных абстракций, присущих операционной системе. Они поддерживают возможность осуществления (псевдо) параллельных операций даже при наличии всего одного центрального процессора. Они превращают один центральный процессор в несколько виртуальных. Без абстракции процессов современные вычисления просто не могут существовать.

Понятие процесса можно трактовать как контейнер ресурсов (выделенная для программ память, файлы, устройства ввода/вывода), а так же как последовательность исполняемых команд.

**Модель процесса**: В этой модели все выполняемое на компьютере программное обеспечение, иногда включая операционную систему, сведено к ряду последовательных процессов, или, для краткости, просто процессов. Процесс — это просто экземпляр выполняемой программы, включая текущие значения счетчика команд, регистров и переменных.



Существуют четыре основных события, приводящих к **созданию процессов**.

* Инициализация системы.
* Выполнение работающим процессом системного вызова, предназначенного для создания процесса. (запуск одним процессом другого)
* Запрос пользователя на создание нового процесса.
* Инициация пакетного задания.

С точки зрения техники всегда один процесс запускает другой.

Фоновые процессы, предназначенные для обработки какой-либо активной деятельности, связанной, например, с электронной почтой, веб-страницами, новостями, выводом информации на печать и т. д., называются демонами.  
В UNIX существует только один системный вызов для создания нового процесса — fork. Этот вызов создает точную копию вызывающего процесса. В Windows все происходит иначе: одним вызовом функции Win32 CreateProcess создается процесс, и в него загружается нужная программа.  
В обеих системах, UNIX и Windows, после создания процесса родительский и дочерний процессы обладают своими собственными, отдельными адресными пространствами.

Рано или поздно новые процессы будут завершены, обычно в силу следующих обстоятельств:

* обычного выхода (добровольно);
* выхода при возникновении ошибки (добровольно);
* возникновения фатальной ошибки (принудительно);
* уничтожения другим процессом (принудительно).

(из лекций:

* штатное завершение(программа отработала)
* ошибка программы, после чего происходит штатное завершение
* программа совершила ошибку и была завершена принудительно (не штатно)
* принудительное завершение (не обязательно из-за ошибки) (не штатно) )

Состояния процесса:

* выполняемый (в данный момент использующий центральный процессор);
* готовый (работоспособный, но временно приостановленный, чтобы дать возможность выполнения другому процессу);
* заблокированный (неспособный выполняться, пока не возникнет какое-нибудь внешнее событие).

(из лекций:

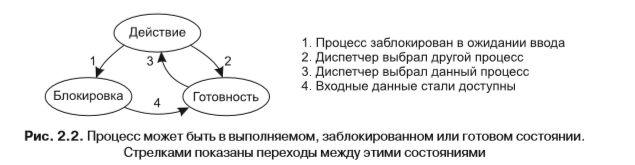
* Исполнения (у процесса есть и данные, и процессор (процессорное время)
* Готовности (данные есть, но процессора не хватает)
* Ожидания (не хватает данных, нет процессора)

Состояние ожидания контролируется ОС (как правило).

При наличии единственного процесса, если тот находится в состоянии ожидания – процессор простаивает. Значит, нужно больше одного процесса.

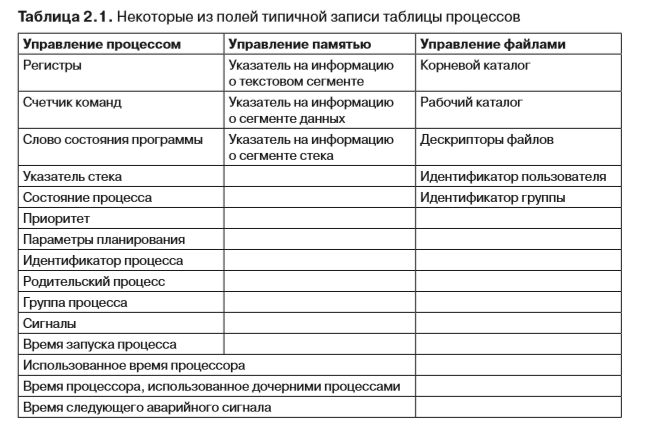
Ожидание->готовность->исполнение

)



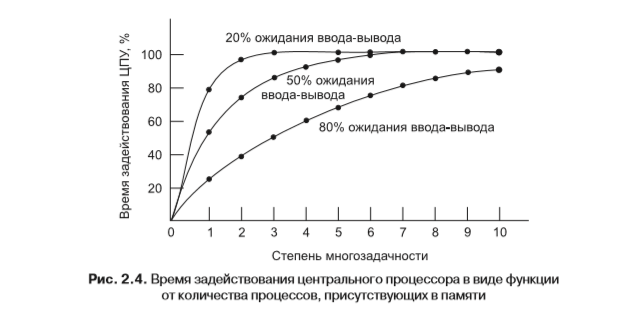
Переход 1 происходит в том случае, если операционная система определит, что процесс в данный момент выполняться не может. В некоторых системах для перехода в заблокированное состояние процесс может осуществить такой системный вызов, как pause.. Переходы 2 и 3 вызываются планировщиком процессов, который является частью операционной системы, без какого-либо оповещения самого процесса. Переход 2 происходит, когда планировщик решит, что выполняемый процесс продвинулся достаточно далеко и настало время позволить другому процессу получить долю рабочего времени центрального процессора. Переход 3 происходит, когда все другие процессы получили причитающуюся им долю времени и настал момент предоставить центральный процессор первому процессу для возобновления его выполнения. Переход 4 осуществляется в том случае, если происходит внешнее событие, ожидавшееся процессом (к примеру, поступление входных данных). Если к этому моменту нет других выполняемых процессов, будет вызван переход 3 и процесс возобновится. В противном случае ему придется немного подождать в состоянии готовности, пока не станет доступен центральный процессор и не придет его очередь.

Для реализации модели процессов операционная система ведет таблицу процессов, в которой каждая запись соответствует какому-нибудь процессу. Эти записи содержат важную информацию о состоянии процесса, включая счетчик команд, указатель стека, распределение памяти, состояние открытых им файлов, его учетную и планировочную информацию и все остальное, касающееся процесса, что должно быть сохранено, когда процесс переключается из состояния выполнения в состояние готовности или блокировки, чтобы позже он мог возобновить выполнение, как будто никогда не останавливался.

Процесс во время своего выполнения может быть прерван тысячи раз, но ключевая идея состоит в том, что после каждого прерывания прерванный процесс возвращается в точности к такому же состоянию, в котором он был до того, как случилось прерывание.

Режим многозадачности позволяет использовать центральный процессор более рационально.

Лучше выстраивать модель на основе вероятностного взгляда на использование центрального процессора. Предположим, что процесс проводит часть своего времени p в ожидании завершения операций ввода-вывода. При одновременном присутствии в памяти n процессов вероятность того, что все n процессов ожидают завершения ввода-вывода (в случае чего процессор простаивает), равна pn. Тогда время задействования процессора вычисляется по формуле Время задействования ценрального процессора = 1 − pn (время задействования центрального процессора в виде функции от аргумента n, который называется степенью многозадачности).



Судя по рисунку, если процесс тратит 80 % своего времени на ожидание завершения ввода-вывода, то для снижения простоя процессора до уровня не более 10 % в памяти могут одновременно находиться по крайней мере 10 процессов (на лекции насчитали 11, и это правильно).

Справедливости ради следует заметить, что рассмотренная нами вероятностная модель носит весьма приблизительный характер. В ней безусловно предполагается, что все n процессов являются независимыми друг от друга, а значит, в системе с пятью процессами в памяти вполне допустимо иметь три выполняемых и два ожидающих процесса. Но имея один центральный процессор, мы не может иметь сразу три выполняемых процесса, поэтому процесс, который становится готовым к работе при занятом центральном процессоре, вынужден ожидать своей очереди. Из-за этого процессы не обладают независимостью.

Несмотря на упрощенность модели, представленной на рис. 2.4, тем не менее она может быть использована для специфических, хотя и весьма приблизительных предсказаний, касающихся производительности центрального процессора. Предположим, к примеру, что память компьютера составляет 8 Гбайт, операционная система и ее таблицы занимают до 2 Гбайт, а каждая пользовательская программа также занимает до 2 Гбайт. Этот объем позволяет одновременно разместить в памяти три пользовательские программы. При среднем ожидании ввода-вывода, составляющем 80 % времени, мы имеем загруженность центрального процессора (если игнорировать издержки на работу операционной системы), равную 1 – 0,83, или около 49 %. Увеличение объема памяти еще на 8 Гбайт позволит системе перейти от трехкратной многозадачности к семикратной, что повысит загруженность центрального процессора до 79 %. Иными словами, дополнительные 8 Гбайт памяти увеличат его производительность на 30 %. Увеличение памяти еще на 8 Гбайт поднимет уровень производительности всего лишь с 79 до 91 %, то есть дополнительный прирост производительности составит только 12 %.