ВЫСШАЯ ШКОЛА ЭКОНОМИКИ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ

И. М. Гостев

ОПЕРАЦИОННЫЕ СИСТЕМЫ

УЧЕБНИК И ПРАКТИКУМ
ДЛЯ АКАДЕМИЧЕСКОГО БАКАЛАВРИАТА

Рекомендовано Учебно–методическим отделом высшего образования в качестве учебника для студентов высших учебных заведений, обучающихся по техническим направлениям

Книга доступна в электронной библиотечной системе biblio-online.ru

Москва ▪ Юрайт ▪ 2016

УДК 004(075.8)
ББК 32.973-018.2я73
Г72

***Автор*:**

Гостев Иван Михайлович — доктор технических наук, доцент, профессор кафедры управления информационными системами и цифровой инфраструктурой Школы бизнес-информатики факультета бизнеса и менеджмента Национального исследовательского университета «Высшая школа экономики».

***Рецензенты*:**

Афанасьев А. П. — доктор физико-математических наук, профессор, заведующий Центром распределенных вычислений Института проблем передачи информации имени А. А. Харкевича Российской академии наук;

Севастьянов Л. А. — доктор физико-математических наук, профессор кафедры прикладной информатики и теории вероятностей Российского университета дружбы народов.

Гостев, И. М.

Г72 Операционные системы : учебник и практикум для академического бакалавриата / И. М. Гостев. — М. : Издательство Юрайт, 2016. — 158 с. — Серия : Бакалавр. Академический курс.

ISBN 978-5-9916-8248-0

В книге рассматриваются наиболее общие вопросы построения операционных систем, независимо от их производителя. Изложены основные принципы управления процессами и организации файловой и сетевой подсистем. Рассмотрены механизмы взаимодействия процессов между собой и пользователем.

Содержание учебника соответствует актуальным требованиям Федерального государственного образовательного стандарта высшего образования.

Книга может быть полезна как для студентов, обучающихся по информационным специальностям, так и всем, кто хочет понять, как организованы операционные системы.

УДК 004(075.8)
ББК 32.973-018.2я73

ISBN 978-5-9916-8248-0

Оглавление

Предисловие

В настоящее время компьютерные науки стремительно развиваются, и отследить все изменения, даже в некоторой довольно узкой области из этого направления, практически невозможно. Начиная читать лекции по операционным системам (ОС) более 15 лет назад, автор столкнулся с несколькими характерными для данной области знаний проблемами.

Во-первых, из-за бурного развития данной области некоторая излагаемая на лекциях информация устаревает к тому моменту, как слушатели закончат вуз. Во-вторых, при изучении операционных систем имеется очень много литературы, и в каждом источнике один и тот же материал представлен разными способами. В-третьих, такое разнообразное изложение материала приводит студентов к затруднению в понимании сути изучаемого предмета.

Поскольку новые версии операционных систем появляются каждые полтора-два года, было принято решение о включении в книгу (и курс обучения) такого материала, который не будет устаревать. Он представляет собой описание некоторых механизмов и архитектуры построения ОС, которые не изменяются и не будут изменяться независимо и времени выпуска и фирмы изготовителя. Иными словами, материал книги представляет собой некоторые наиболее общие принципы построения операционных систем, которые были разработаны более 50 лет назад и практически не изменились за прошедшее время.

В настоящее время активно рекламируются вроде бы новые решения, например такие, как новомодные «облачные» технологии или Big data, но с точки зрения алгоритмики и технологии построения операционных и (или) вычислительных систем в них, кроме маркетингового хода, нет ничего нового. Подавляющее большинство известных на сегодня алгоритмов были разработаны в конце 1950-х — начале 1960-х гг. Например, появляющиеся в настоящее время такие решения, как бесконтактное управление компьютером с помощью жестов, в те годы из-за низкого быстродействия вычислительных машин было технически не реализуемо.

В современном компьютерном мире фактически доминируют и конкурируют две операционные системы — Unix (Linux) и Microsoft Windows (MS). Первая со всеми ее разновидностями всегда была открытой и прозрачной по коду и архитектуре, независимо и фирмы изготовителя или программистского сообщества (для Linux). Вторая же полностью закрыта и нет никаких гарантий, что опубликованные описания ОС от MS соответствуют действительности. Огромная конкуренция в области вычислительных средств, а в последнее время и повышение требований к секретности информации приводят к тому, что продукты MS законодательно запрещено применять в государственных учреждениях в ряде стран.

Аналогичная ситуация назревает и в России. Для решения задач по сохранению секретности и устранению вредных «закладок» были разработаны отечественные операционные системы на основе открытого кода Unix (Linux), такие как «Патриот ОС», «Заря», «Astra Linux», «ALT Linux», «Роса», «Эльбрус», которые успешно заменяют ОС MS во многих отраслях промышленности России. После подтверждения фирмой MS того, что 10-я версия Windows несанкционированно отправляет данные с компьютеров пользователей в аналитические центры MS, вопрос по импортозамещению иностранных ОС стал еще острее. Так, в некоторых областях России уже приняты законодательные решения об отказе дальнейшего использования продуктов компании MS. Исходя из вышеизложенного актуальность изучения архитектуры операционных систем, особенно на базе Unix (Linux), только возрастает.

Представленный материал предназначен как для самостоятельного изучения, так и для закрепления материала, полученного на лекциях. Содержание книги можно использовать на различных стадиях изучения ОС. Для начального обучения или для изучения ОС студентами нетехнических специальностей параграфы, связанные с функциями интерфейсов подсистем ядра, можно опустить. При более детальном изучении кроме функций интерфейса и структур данных рекомендуется проведение лабораторных работ по изучению команд Unix, а также написание небольших тестирующих программ по взаимодействию с основными подсистемами ядра. Для этого в приложениях приведены краткая система команд интерпретатора Bash для Unix и перечень вариантов заданий. При более серьезном изучении возможно написание программ на языке Си по взаимодействию пользователя с ОС на основе функций интерфейса.

Издание предназначено для изучения в бакалавриате для специальностей, в которых необходимо знать и понимать архитектуру IT-систем (IT — Information Technology, информационные технологии), которая изложена здесь на основе ОС, таких как компьютерные науки, бизнес-информатика, прикладная математика и другие технические специальности, где необходимо понимание работы ОС. В случае гуманитарных специальностей книга может быть рекомендована для дополнительного изучения в магистратуре, как источник по внутренней организации ОС.

Для упрощения понимания материала и его сопоставления с иностранной литературой для используемых терминов дается перевод на английский язык.

Целью изучения данной дисциплины является формирование у обучающегося целостной концептуальной модели ОС со знанием основных принципов ее функционирования; пониманием принципов конструирования ее внутренней архитектуры; функциональным представлением ее составляющих подсистем и их взаимодействием.

Задачами дисциплины являются получение обучающимися твердых теоретических знаний об устройстве операционных систем и понимание механизмов функционирования процессов в них, на основе которых можно выбрать нужную ОС, настроить ее под решение конкретных задач, устранить неисправности функционирования и получить некоторые навыки по написанию программ для взаимодействия пользователя с ОС.

В зависимости от обстоятельности изучения материала к теоретическому рассмотрению можно добавить практические работы по отработке команд Unix.

В результате изучения данной книги студент должен:

знать

- архитектурные принципы и методологию построения ОС;
- основные функциональные компоненты ОС;
- алгоритмы функционирования компонентов в ОС;
- принципы взаимодействия компонентов в ОС;

уметь

- администрировать и конфигурировать ОС под свои потребности;
- анализировать состояние ОС по характеру протекающих в ней процессов;
- организовать взаимодействие с ОС на программном уровне для решения конкретных задач;
- применять принципы и алгоритмы работы функциональных компонентов ОС в своей производственной деятельности;

владеть

- навыками решения задач по конфигурированию и настройке ОС Unix-подобных систем;
- диагностирования состояния ОС по анализу исполнения процессов в ней;
- программирования на командном языке, в том числе и в режиме удаленного доступа;
- программирования на языке высокого уровня для решения системных задач.

Глава 1
ВВЕДЕНИЕ В ОПЕРАЦИОННЫЕ СИСТЕМЫ

Что такое операционная система? Это очень обширное понятие, касающееся всей совокупности программных кодов, которые дают возможность функционировать процессору и другим компонентам и устройствам вычислительной системы.

Операционные системы появляются тогда, когда возникает необходимость управления процессором или его аналогом. Они существуют и в таких простых системах, как автомат для продажи баночного пива, и в системах управления полетом космических объектов. Соответственно и сложность таких ОС зависит от сложности устройства, которым необходимо управлять. Следовательно, разновидности этих систем и их функции могут быть различными. Однако имеются функции, по которым можно безошибочно определить существование операционной системы. К ним относятся:

- управление процессором;
- управление всеми устройствами (компонентами) в системе (внешними и внутренними);
- управление данными и потоками данных в системе;
- синхронизация работы всех устройств и процессов в системе;
- обработка прерываний (внешних и внутренних);
- предоставление пользователю интерфейсов для управления системой.

В данной книге ОС будет рассмотрена именно с этой точки зрения и особое внимание будет уделено функциям, которые она выполняет, и механизмам, на основании которых эти функции реализованы. Принципы работы ОС здесь рассматривается независимо от ее конкретной реализации (Unix, MacOC, Windows). Все изложенные в книге механизмы функционирования ОС присутствуют в той или иной степени во всех существующих ОС, но могут иметь некоторые отличия в конкретной реализации.

В настоящей книге не рассматриваются графические, тактильные и голосовые интерфейсы, поскольку: во-первых, они практически каждый год изменяются, а во-вторых, представляют собой надстройку любой ОС, и к архитектуре и механизмам работы ОС не имеют отношения.

1.1. Классификация операционных систем

Классификация операционных систем имеет множество способов. По одной из таких таксономий все операционные системы разделяются на *системы реального времени* и *системы разделения времени*.

Операционная система *реального времени* характеризуется тем, что ее функционирование определено внешними запросами, поступающими в заранее не определенное время. Последний запрос всегда имеет наивысший приоритет выполнения. Это означает, что все остальные задачи, которые были в системе, откладываются, и начинается обработка вновь поступившего запроса. Обработка каждого запроса имеет жесткие временные рамки. Для реализации такой системы необходимо иметь высокопроизводительную вычислительную систему, скорость обработки заданий в которой выше, чем темп поступающих в систему запросов.

Примером такой системы является система противовоздушной обороны, когда время обнаружения и уничтожения быстролетящего объекта составляет доли секунды. Здесь запросом к системе является появление на экране радара некоторого летящего объекта. Обработка запроса означает распознавание объекта и, если это объект «чужой», выполнение действий по его уничтожению. Очевидно, что если не распознать летающий объект за некоторое время (время его полета), то сама система может быть уничтожена. Расчет характеристик такой системы производится исходя из того, какое максимальное количество запросов необходимо обслужить за некоторое небольшое время.

Другим примером ОС реального времени является система управления устройствами в автомобиле. В покое ОС рассчитывает и обеспечивает максимально экономичный режим движения, учитывая показания датчиков (температуры, скорости движения, влажности, ABS и т.п.). При торможении автомобиля ОС переключается на режим максимально плавного торможения и минимального пути торможения.

В настоящее время поддерживаются несколько ОС реального времени, например LynxOS, QNX, VxWorks.

Совсем другая стратегия обслуживания запросов существует в системах *разделения времени*. В этих системах постулировано, что каждая задача за некоторое время должна иметь доступ к центральному процессору. Иными словами, в таких ОС существует очередь задач, в которой каждая задача выполняется небольшое, но всегда гарантированное время за некоторой малый промежуток времени, называемый квантом. Сколько бы задач ни находилось в системе одновременно, все они будут выполняться в течение некоторого кванта. Очевидно, что при фиксированных ресурсах системы время выполнения каждой задачи в системе с ростом числа задач будет возрастать. Тем не менее каждая задача всегда имеет доступ к ресурсам центрального процессора. Для обеспечения этого механизма имеются различные средства. Это может быть и кэширование, и система приоритетов задач, и механизмы свопинга и пейджинга, и различные оптимизационные решения по обмену информации между модулями системы, и т.п.

Характерные отличия ОС разделения времени и ОС реального времени приведены в табл. 1.1.

По другой классификации все ОС принято разделять по их функциональному составу.

1. ***Монолитные ОС***, т.е. ОС, состоящие из модулей, функционирование которых невозможно представить раздельно друг от друга и тем более сгруппировать в уровни. Характерным примером такой системы является Unix.

Таблица 1.1

Сравнение характеристик ОС

Характеристика	ОС реального времени	ОС разделения времени
Время реагирования на внешний запрос	Минимальное. Последний запрос обрабатывается первым	Не более кванта, который обычно равен 0,01 с
Эффективность работы системы	Не имеет значения	Существенна
Размер решаемых задач	Не должен быть слишком большой	Не ограничен
Время выполнения задачи	Суммарное время выполнения всех задач в системе должно быть меньше, чем интервал времени между поступлением задач в систему	Определяется сложностью самой задачи
Быстродействие системы	Должно быть максимально возможным для темпа поступления запросов	Особой роли не играет
Гарантии обработки задачи до конца за определенное время	Должна быть 100%-я гарантия. Иначе система не работоспособна	Не существует
Инструментарий для разработки программ	Отсутствует	Широкий выбор средств разработки, отладки и тестирования программ
Интерфейс с пользователем	Как правило, примитивный интерфейс, обычно в виде командной строки	Богатый выбор интерфейсов, начиная от командной строки и заканчивая утонченными оконными интерфейсами

2. ***Уровневое представление*** **ОС**, основано на представленные отдельными уровнями, которые могут быть вертикальными и горизонтальными. При вертикальной организации уровней (рис. 1.1, *б*) каждому уровню соответствует определенная функция. Уровень включает в себя все программные компоненты, необходимые для ее выполнения, включая драйверы физических устройств. В такой системе копирование с диска на экран и с одного диска на другой — это совершенно разные уровни. Преимущество такой системы заключается в быстроте выполнения операций, поскольку каждая из них ориентирована только на одно устройство. Каждая операция должна содержать, кроме собственно самого функционала, еще и драйверы тех устройств, над которыми и будет производиться операция. Существенным недостатком такой системы являются большое количество уровней и необходимость в разработке для каждого нового устройства полного набора уровней, обеспечивающих работу с ним. Операционные системы такой архитектуры обычно разрабатываются и используются в технологических процессах, там, где количество команд и устройств ограничено.

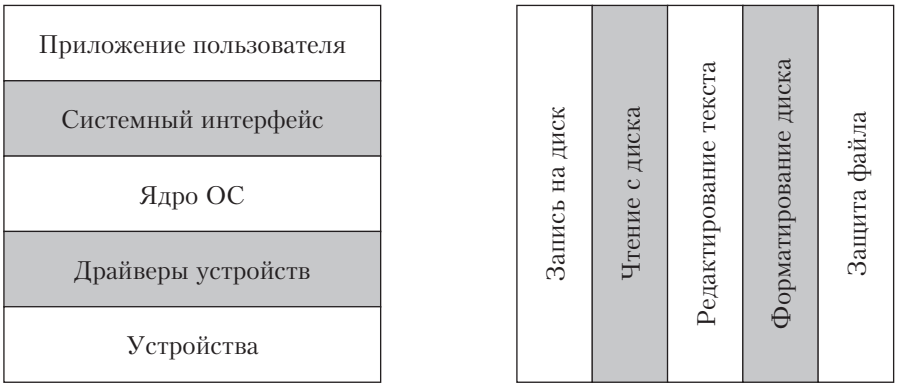

Рис. 1.1. **Организация ОС**:
а — горизонтальная; *б* — вертикальная

В другом типе ОС — с горизонтальным расположением уровней (см. рис. 1.1, *а*) — функции ОС разделены между уровнями. На верхних уровнях сосредоточены функции интерфейса с пользователем, а на нижних — драйверы, поддержка работы физических устройств. В таких системах добавление функциональности в некоторый уровень или расширение поддержки новых устройств достигается добавлением нового драйвера и не затрагивает другие уровни. Таким образом, достигаются бóльшая универсальность и меньшая трудоемкость в создании и модификации подобных систем. Характерным примером такой ОС является MS DOS.

Уровневое представление — наиболее часто интерпретируемая организация ОС, независимо от ее архитектуры. Однако она не всегда может дать настоящее представление о взаимодействии различных ее составляющих.

3. ***Виртуальная ОС*** — это программная реализация, эмулирующая аппаратное и (или) программное обеспечение некоторой платформы. Помимо процессора, виртуальная ОС может эмулировать работу как отдельных компонентов аппаратного обеспечения, так и полностью реального компьютера. Хорошим примером такой эмуляции служит реализация ОС автомобиля, использующей специальный процессор на другой платформе, например Intel и ОС Windows. Здесь происходит эмуляция как аппаратного (бортовой компьютер и датчики), так и программного обеспечения (собственно сама ОС). При работе виртуальной ОС происходит выделение определенных ресурсов из основной системы для выполнения некоторых программ, которые невозможно выполнить в основной ОС (например, при несовпадении системы команд процессоров).

Область применения виртуальных ОС весьма значительна — от выполнения функций по тестированию ПО узкоспециализированных ОС до формирования больших виртуальных компьютерных сетей, которые легко создавать, масштабировать, контролировать и в которых несложно распределять нагрузку.

В последнее время часто используется практика установки на реальный компьютер несколько ОС, например, из-под ОС Windows запускать в вир-

туальной машине ОС Linux или наоборот. Более того, на одном компьютере одновременно можно запустить несколько виртуальных ОС, которые могут выступать как серверы или рабочие станции с целью оптимизации работы такой виртуальной сети.

В качестве подобных виртуальных ОС имеется большое количество продуктов, позволяющих эмулировать одну операционную систему внутри другой, например VirtualBox, VMware, Windows VirtualPC, Parallels Desktop и т.д. Особую роль в последнее время в связи с развитием ОС смартфонов стали играть эмуляторы для ОС Android и IOS, которые позволяют разрабатывать программные приложения к ним и тестировать их, например XCode, Droid4X, AMIDuOS, BlueStacks Andy, Windroy и др.

Виртуальные ОС могут быть использованы:

- для разработки новых аппаратно-программных платформ, на которых реализация компиляторов невозможна или нецелесообразна;
- защиты основной ОС при тестировании и (или) исследования некоторой системы, запуск которой может повредить основную ОС (запуск программ в «песочнице»);
- исследования работоспособности и производительности некоторого специального ПО, например игровых приставок;
- изучения поведения компьютерных вирусов в изолированной среде;
- исследования работоспособности виртуальных ОС при их переносе с одного компьютера на другой при построении сложных кластеров;
- перенесения одного программного приложения без каких-либо изменений с машин одной архитектуры и ОС на другие, например, как это было сделано с VJM (Virtual Java Machines), которая в настоящее время имеется в любой ОС.

4. ***Микроядерная архитектура.*** В таких ОС существует центральный модуль, представляющий собой супервизорную часть ОС. Из этого модуля удалено «все, что можно удалить», а функции сокращены до предела. Из исполняемых функций обычно оставляют только управление виртуальной памятью, поддержку процессов и потоков, межпроцессное взаимодействие (IPC), управление прерываниями и некоторые сервисы процессора.

Микроядро — наиболее приоритетная часть ОС — передает управление на другие модули ОС для выполнения определенных операций, например операций ввода-вывода.

Одним из представителей микроядерных ОС является ОС РВ QNX. В функции ее микроядра входят только диспетчеризация процессов, IPC, обработка прерываний и некоторые сетевые сервисы. В такое микроядро заложено небольшое число системных вызовов, и его можно разместить полностью в кэше процессора Intel 486, так как его размер не превышает 46 Кбайт. Для построения минимальной системы QNX к ядру необходимо добавить менеджеры процессов, файловой системы и устройств, которые исполняются вне пространства ядра.

К микроядерным архитектурам относят также MAC OC X, Symbian OS, Jari OS.

5. ***Операционная система Клиент-сервер.*** Расширяя понятие микроядра, в Microsoft была разработана ОС Windows NT, в которой функции

управления процессором были выделены в отдельный модуль для реализации поддержки различных архитектур процессоров. В то же время был разработан единый механизм взаимодействия процессов пользователя с ядром[1]. При этом в архитектуре Windows NT на ядро ОС возложены все обычные функции, такие как управление памятью, диспетчеризацией, безопасностью, вводом-выводом и межпроцессорными обменами. Кроме ядра NT содержит специальный слой, названный *уровнем абстракции от оборудования* (HAL — Hardware Abstraction Layer), который изолирует ядро, драйверы устройств и исполняемую часть NT от аппаратных платформ, на которых должна работать ОС. Такое решение позволяет, не изменяя всего остального программного обеспечения, переходить с одной аппаратной платформы к другой. Более того, заменяя «драйвер» процессора, можно легко перейти от одно- к многопроцессорной системе.

Для этой системы был разработан специальный механизм защиты, известный как процедуры удаленного вызова RPC (Remote Procedure Call), организующий работу пользователя отдельно от самой операционной системы аналогично работе удаленного клиента. Приложение работает в собственном пространстве памяти с загруженным в него набором системных функций, необходимых для его работы. Клиент взаимодействует только с интерфейсами, исполнение которых производится в отдельном адресном пространстве каждого процесса.

Кроме временного и архитектурного представления существует еще много градаций и форм представлений операционных систем, например, на основе совокупности операций (функциональный подход) или потоков данных и потоков управления и т.п. Мы их рассматривать не будем.

1.2. Процессы в операционной системе

1.2.1. Процессы и примитивы

Программы ОС можно разделить на программы, *расширяющие функции аппаратуры*, и программы, по сути являющиеся *обслуживающими*. Для того чтобы установить, к какому классу относится программа, необходимо знать, каким образом ей передается управление. На основании этой предпосылки традиционно все программы разделяются на процессы и примитивы[2].

Существует несколько способов передачи управления программам. Любая программа, состоящая из нескольких ветвей, передает управление на них с помощью команд перехода. Эта передача никак не отражается на других объектах ОС. Команда перехода не влечет за собой изменений ни в системных очередях, ни в специальных таблицах.

Более сложный, строгий и формализованный способ обращения одной программы к другой — вызов подпрограммы. Здесь для передачи параме-

---

[1] *Solomon D. A.* The Windows NT Kernel Architecture // IEEE Computer. October. 1998. P. 40—47.

[2] *Бах М. Дж.* Архитектура операционной системы UNIX : пер. с англ. А. В. Крюкова. Copyright, 1986.