Понятие операционной системы. Виртуальные машины

Современный компьютер – сложнейшая аппаратно-программная система. Написание программ для компьютера, их отладка и последующее выполнение представляет собой сложную трудоемкую задачу. Основная причина этого – огромная разница между тем, что удобно для людей, и тем, что удобно для компьютеров. Компьютер понимает только свой, машинный язык (назовем его Я0), а для человека наиболее удобен разговорный или хотя бы язык описания алгоритмов – алгоритмический язык. Проблему можно решить двумя способами. Оба способа связаны с разработкой команд, которые были бы более удобны для человека, чем встроенные машинные команды компьютера. Эти новые команды в совокупности формируют некоторый язык, который назовем Я1.

Упомянутые два способа решения проблемы различаются тем, каким образом компьютер будет выполнять программы, написанные на языке Я1. Первый способ – замена каждой команды языка Я1 на эквивалентный набор команд в языке Я0. В этом случае компьютер выполняет новую программу, написанную на языке Я0, вместо программы, написанной на языке Я1. Эта технология называется *трансляцией*.

Второй способ – написание программы на языке Я0, которая берет программы, написанные на языке Я1, в качестве входных данных, рассматривает каждую команду по очереди и сразу выполняет эквивалентный набор команд языка Я0. Эта технология не требует составления новой программы на Я0. Она называется *интерпретацией*, а программа, которая осуществляет интерпретацию, называется *интерпретатором*.

В подобной ситуации проще представить себе существование гипотетического компьютера или *виртуальной* машины, для которой машинным языком является язык Я1, чем думать о трансляции и интерпретации. Назовем такую виртуальную машину М1, а виртуальную машину с языком Я0 – М0. Для виртуальных машин можно будет писать программы, как будто они (машины) действительно существуют.

Очевидно, можно пойти дальше – создать еще набор команд, который в большей степени ориентирован на человека и в меньшей степени на компьютер, чем Я1. Этот набор формирует язык Я2 и, соответственно, виртуальную машину М2. Так можно продолжать до тех пор, пока не дойдем до подходящего нам языка уровня n.

Большинство современных компьютеров состоит из двух и более уровней. Уровень 0 – аппаратное обеспечение машины. Электронные схемы этого уровня выполняют программы, написанные на языке уровня 1. Следующий уровень – *микроархитектурный* уровень.

На этом уровне можно видеть совокупности 8 или 32 (иногда и больше) регистров, которые формируют локальную память и АЛУ (арифметико-логическое устройство). Регистры вместе с АЛУ формируют тракт данных, по которому поступают данные. Основная операция этого тракта заключается в следующем. Выбирается один или два регистра, АЛУ производит над ними какую-то операцию, а результат помещается в один из этих регистров. На некоторых машинах работа тракта контролируется особой программой, которая называется микропрограммой. В других машинах такой контроль выполняется аппаратным обеспечением.

Следующий (второй) уровень составляет уровень *архитектуры системы команд*. Команды используют регистры и другие возможности аппаратуры. Команды формируют уровень ISA (Instruction Set Architecture), называемый машинным языком. Обычно

машинный язык содержит от 50 до 300 команд, служащих преимущественно для перемещения данных по компьютеру, выполнения арифметических операций и сравнения величин.

Следующий (третий) уровень обычно – гибридный. Большинство команд в его языке есть также и на уровне архитектуры системы команд. У этого уровня есть некоторые дополнительные особенности: набор новых команд, другая организация памяти, способность выполнять две и более программы одновременно и некоторые другие. С течением времени набор таких команд существенно расширился. В нем появились так называемые макросы операционной системы или вызовы супервизора, называемые теперь системными вызовами.

Новые средства, появившиеся на третьем уровне, выполняются интерпретатором, который работает на втором уровне. Этот интерпретатор был когда-то назван *операционной системой*. Команды третьего уровня, идентичные командам второго уровня, выполняются микропрограммой или аппаратным обеспечением, но не операционной системой. Иными словами, одна часть команд третьего уровня интерпретируется операционной системой, а другая часть – микропрограммой. Вот почему этот уровень операционной системы считается гибридным.

Операционная система была создана для того, чтобы автоматизировать работу оператора и скрыть от пользователя сложности общения с аппаратурой, предоставив ему более удобную систему команд. Нижние три уровня (с нулевого по второй) конструируются не для того, чтобы с ними работал обычный программист. Они изначально предназначены для работы интерпретаторов и трансляторов, поддерживающих более высокие уровни. Эти трансляторы и интерпретаторы составляются системными программистами, которые специализируются на разработке и построении новых виртуальных машин.

Над операционной системой (ОС) расположены остальные системные программы. Здесь находятся интерпретатор команд (оболочка), компиляторы, редакторы и т.д. Подобные программы не являются частью ОС (иногда оболочку пользователи считают операционной системой). Под операционной системой обычно понимается то программное обеспечение, которое запускается в *режиме ядра* или, как еще его называют, *режиме супервизора*. Она защищена от вмешательства пользователя с помощью специальных аппаратных средств.

Четвертый уровень представляет собой символическую форму одного из языков низкого уровня (обычно ассемблер). На этом уровне можно писать программы в приемлемой для человека форме. Эти программы сначала транслируются на язык уровня 1, 2 или 3, а затем интерпретируются соответствующей виртуальной или фактически существующей (физической) машиной.

Уровни с пятого и выше предназначены для прикладных программистов, решающих конкретные задачи на языках высокого уровня (C, C++, C#, VBA и др.). Компиляторы и редакторы этих уровней *запускаются в пользовательском режиме*. На еще более высоких уровнях располагаются прикладные программы пользователей.

Большинство пользователей компьютеров имеют опыт общения с операционной системой, по крайней мере, в той степени, чтобы эффективно выполнять свои текущие задачи. Однако они испытывают затруднения при попытке дать определение операционной системе. В известной степени проблема связана с тем, что операционные

системы выполняют две основные, но практически не связанные между собой функции: расширение возможностей компьютера и управление его ресурсами.

С точки зрения пользователя ОС выполняет функцию расширенной машины или виртуальной машины, в которой легче программировать и легче работать, чем непосредственно с аппаратным обеспечением, составляющим реальный компьютер. Операционная система не только устраняет необходимость работы непосредственно с дисками и предоставляет простой, ориентированный на работу с файлами интерфейс, но и скрывает множество неприятной работы с прерываниями, счетчиками времени, организацией памяти и другими компонентами низкого уровня.

Однако концепция, рассматривающая операционную систему прежде всего как удобный интерфейс пользователя, – это взгляд сверху вниз. Альтернативный взгляд, снизу вверх, дает представление об операционной системе как о механизме, присутствующем в компьютере для управления всеми компонентами этой сложнейшей системы. В соответствии с этим подходом работа операционной системы заключается в обеспечении организованного и контролируемого распределения процессоров, памяти, дисков, принтеров, устройств ввода-вывода, датчиков времени и т.п. между различными программами, конкурирующими за право их использовать.

1.2. Операционная система, среда и операционная оболочка

Операционные системы (ОС) в современном их понимании (их назначении и сущности) появились значительно позже первых компьютеров (правда, по всей видимости, и исчезнут в этой сущности в компьютерах будущего). Почему и когда появились ОС? Считается[1] что первая цифровая вычислительная машина ENIAC (Electronic Numerical Integrator and Computer) была создана в 1946 году по проекту "Проект PX" Министерства обороны США. На реализацию проекта затрачено 500 тыс. долларов. Компьютер содержал 18000 электронных ламп, массу всякой электроники, включал в себя 12 десятиразрядных сумматоров, а для ускорения некоторых арифметических операций имел умножитель и "делитель-извлекатель" квадратного корня. Программирование сводилось к связыванию различных блоков проводами. Конечно, никакого программного обеспечения и тем более операционных систем тогда еще не существовало.

Интенсивное создание различных моделей ЭВМ относится к началу 50-х годов прошлого века. В эти годы одни и те же группы людей участвовали и в проектировании, и в создании, и в программировании, и в эксплуатации ЭВМ. Программирование осуществлялось исключительно на машинном языке (а затем на ассемблере), не было никакого системного программного обеспечения, кроме библиотек математических и служебных подпрограмм. Операционные системы еще не появились, а все задачи организации вычислительного процесса решались вручную каждым программистом с примитивного пульта управления ЭВМ.

С появлением полупроводниковых элементов вычислительные возможности компьютеров существенно выросли. Наряду с этим заметно прогрессировали достижения в области автоматизации программирования и организации вычислительных работ. Появились алгоритмические языки (алгол, фортран, кобол) и системное программное обеспечение (трансляторы, редакторы связи, загрузчики и др.). Выполнение программ усложнилось и включало в себя следующие основные действия:

- загрузка нужного транслятора (установка нужных МЛ и др.);
- запуск транслятора и получение программы в машинных кодах;

- связывание программы с библиотечными подпрограммами;
- загрузка программы в оперативную память;
- запуск программы;
- вывод результатов работы программы на печатающее или другое периферийное устройство.

Для организации эффективной загрузки всех средств компьютера в штаты вычислительных центров ввели должности специально обученных операторов, профессионально выполнявших работу по организации вычислительного процесса для всех пользователей этого центра. Однако, как бы ни был подготовлен оператор, ему тяжело состязаться в производительности с работой устройств компьютера. И поэтому большую часть времени дорогостоящий процессор простаивал, а следовательно, использование компьютеров не было эффективным.

С целью исключения простоев были предприняты попытки разработки специальных программ – мониторов, прообразов первых операционных систем, которые осуществляли автоматический переход от задания к заданию. Считается, что первую операционную систему создала в 1952 году для своих компьютеров IBM-701 исследовательская лаборатория фирмы General Motors. В 1955 году эта фирма и North American Aviation совместно разработали ОС для компьютера IBM-704.

В конце 50-х годов прошлого века ведущие фирмы изготовители поставляли операционные системы со следующими характеристиками:

- пакетная обработка одного потока задач;
- наличие стандартных программ ввода-вывода;
- возможности автоматического перехода от программы к программе;
- средства восстановления после ошибок, обеспечивающие автоматическую "очистку" компьютера в случае аварийного завершения очередной задачи и позволяющие запускать следующую задачу при минимальном вмешательстве оператора;
- языки управления заданиями, предоставляющие пользователям возможность описывать свои задания и ресурсы, требуемые для их выполнения.

Пакет представляет собой набор (колоду) перфокарт, организованную специальным образом (задание, программы, данные). Для ускорения работы он мог переноситься на магнитную ленту или диск. Это позволяло сократить простои дорогой аппаратуры. Надо сказать, что в настоящее время в связи с прогрессом микроэлектронных технологий и методологий программирования значительно снизилась стоимость аппаратных и программных средств компьютерной техники. Поэтому сейчас основное внимание уделяется тому, чтобы сделать работу пользователей и программистов более эффективной, поскольку затраты труда квалифицированных специалистов сейчас представляют собой гораздо большую долю общей стоимости вычислительных систем, чем аппаратные и программные средства компьютеров.

Расположение операционной системы в иерархической структуре программного и аппаратного обеспечения компьютера можно представить, как показано на рис. 1.1.

Рис. 1.1. Иерархическая структура программно-аппаратных средств компьютера

Самый нижний уровень содержит различные устройства компьютера, состоящие из микросхем, проводников, источников питания, электронно-лучевых трубок и т.п. Этот уровень можно разделить на подуровни, например контроллеры устройств, а затем сами устройства. Возможно деление и на большее число уровней. Выше расположен микроархитектурный уровень, на котором физические устройства рассматриваются как отдельные функциональные единицы.

На микроархитектурном уровне находятся внутренние регистры центрального процессора (их может быть несколько) и арифметико-логические устройства со средствами управления ими. На этом уровне реализуется выполнение машинных команд. В процессе выполнения команд используются регистры процессора и устройств, а также другие возможности аппаратуры. Команды, видимые для работающего на ассемблере программиста, формируют уровень ISA (Instruction Set Architecture – архитектура системы команд), часто называемый машинным языком.

Операционная система предназначена для того, чтобы скрыть все эти сложности. Конечный пользователь обычно не интересуется деталями устройства аппаратного обеспечения компьютера. Компьютер ему видится как набор приложений. Приложение может быть написано программистом на каком-либо языке программирования. Для упрощения этой работы программист использует набор системных программ, некоторые из которых называются утилитами. С их помощью реализуются часто используемые функции, которые помогают работать с файлами, управлять устройствами ввода-вывода и т.п. Программист применяет эти средства при разработке программ, а приложения во время выполнения обращаются к утилитам для выполнения определенных функций. Наиболее важной из системных программ является операционная система, которая освобождает программиста от необходимости глубокого знания устройства компьютера и представляет ему удобный интерфейс для его использования. Операционная система выступает в роли посредника, облегчая программисту, пользователям и программным приложениям доступ к различным службам и возможностям компьютера [10].

Таким образом, *операционная система* – это набор программ, контролирующих работу прикладных программ и системных приложений и исполняющих роль интерфейса между пользователями, программистами, прикладными программами, системными приложениями и аппаратным обеспечением компьютера.

Образно можно сказать, что аппаратура компьютера предоставляет "сырую" вычислительную мощность, а задача операционной системы заключается в том, чтобы сделать использование этой вычислительной мощности доступным и по возможности удобным для пользователя. Программист может не знать детали управления конкретными ресурсами (например, диском) компьютера и должен обращаться к операционной системе с соответствующими вызовами, чтобы получить от нее необходимые сервисы и функции. Этот набор сервисов и функций и представляет собой операционную среду, в которой выполняются прикладные программы.

Таким образом, *операционная среда* – это программная среда, образуемая операционной системой, определяющая интерфейс прикладного программирования (API) как множество системных функций и сервисов (системных вызовов), которые предоставляются прикладным программам. Операционная среда может включать несколько интерфейсов прикладного программирования. Кроме основной операционной среды, называемой естественной (native), могут быть организованы путем эмуляции (моделирования) дополнительные программные среды, позволяющие выполнять приложения, которые рассчитаны на другие операционные системы и даже другие компьютеры.

Еще одно важное понятие, связанное с операционной системой, относится к реализации пользовательских интерфейсов. Как правило, любая операционная система обеспечивает удобную работу пользователя за счет средств пользовательского интерфейса. Эти средства могут быть неотъемлемой частью операционной среды (например, графический интерфейс Windows или текстовый интерфейс командной строки MS DOS), а могут быть реализованы отдельной системной программой – оболочкой операционной системы (например, Norton Commander для MS DOS). В общем случае под *оболочкой операционной системы* понимается часть операционной среды, определяющая интерфейс пользователя, его реализацию (текстовый, графический и т.п.), командные и сервисные возможности пользователя по управлению прикладными программами и компьютером.