**№ занятия: 1. Лекция № 1.**

**Тема: Назначение и функции операционных систем.**

Данная лекция открывает тему «Теоретические основы построения операционных систем». На сегодняшний день операционная система является основным элементом построения автоматизированных систем управления. Для лучшего понимания основ построения операционных систем изучим историю появления и развития операционных систем, функциональные компоненты операционной системы и требования к современным операционным системам.

**Учебные вопросы (основная часть):**

**Вопрос 1. Эволюция операционных систем**.

Впервые идея компьютера была предложена английским математиком *Чарльзом Бэбиджем* (*Charles Babage*) в середине девятнадцатого века. Его механическая «аналитическая машина» так и не смогла по-настоящему заработать, потому что технологии того времени не удовлетворяли требованиям, необходимым для изготовления нужных деталей точной механики. Конечно, никакой речи об операционной системе (ОС) для этого «компьютера» не шло.

Настоящее рождение цифровых вычислительных машин произошло вскоре после окончания Второй мировой войны. В середине 1940-х годов в США и Европе (Англия) примерно одновременно были созданы первые ламповые вычислительные устройства. Чуть позже, в 1951 году, была создана первая электронная вычислительная машина и в СССР.

В то время одна и та же группа людей участвовала и в проектировании, и в эксплуатации, и в программировании вычислительной машины. Это была скорее научно-исследовательская работа в области вычислительной техники, а не использование компьютеров в качестве инструмента решения каких-либо практических задач из других прикладных областей. Программирование осуществлялось исключительно на машинном языке. Не было никакого системного программного обеспечения, кроме библиотек математических и служебных подпрограмм, которые программист мог использовать для того, чтобы не писать каждый раз коды, вычисляющие значение какой-либо математической функции или управляющие стандартным устройством ввода/вывода. Операционные системы все еще не появились, все задачи организации вычислительного процесса решались вручную каждым программистом с пульта управления, представляющего собой примитивное устройство ввода/вывода, состоящее из кнопок, переключателей и индикаторов.

С середины 1950-х годов начался следующий период в развитии вычислительной техники, связанный с появлением новой технической базы – полупроводниковых элементов. Выросло быстродействие процессоров, увеличились объемы оперативной и внешней памяти. Компьютеры стали более надежными, они могли непрерывно работать настолько долго, чтобы на них можно было возложить выполнение действительно практически важных задач.

Наряду с совершенствованием аппаратуры заметный прогресс наблюдался в области автоматизации программирования и организации вычислительных работ. В эти годы появились первые алгоритмические языки, и, таким образом, к библиотекам математических и служебных подпрограмм добавился новый тип системного программного обеспечения – трансляторы.

Выполнение каждой программы стало включать большое количество вспомогательных работ: загрузка нужного транслятора (АЛГОЛ, ФОРТРАН, КОБОЛ, и т. п.), запуск транслятора и получение результирующей программы в машинных кодах, связывание программы с библиотечными подпрограммами, загрузка программы в оперативную память, запуск программы, вывод результатов на периферийное устройство. Для организации эффективного совместного использования трансляторов, библиотечных программ и загрузчиков в штат многих вычислительных центров были введены должности операторов, профессионально выполнявших работу по организации вычислительного процесса для всех пользователей этого центра.

Но как бы быстро и надежно, ни работали операторы, они никак не могли состязаться в производительности с работой компьютера, процессор которого большую часть времени простаивал в ожидании, пока оператор запустит очередную задачу. А поскольку процессор представлял собой весьма дорогое устройство, то низкая эффективность его использования означала низкую эффективность использования компьютера в целом. Для решения этой проблемы были разработаны первые системы пакетной обработки, которые автоматизировали всю последовательность действий оператора по организации вычислительного процесса.

Ранние системы пакетной обработки явились прообразом современных операционных систем, они стали первыми системными программами, предназначенными не для обработки данных, а для управления вычислительным процессом. В ходе реализации систем пакетной обработки был разработан формализованный язык управления заданиями, с помощью которого программист сообщал системе и оператору, какие действия и в какой последовательности он хочет выполнить на вычислительной машине. Типовой набор директив обычно включал признак начала отдельной работы, вызов транслятора, вызов загрузчика, признаки начала и конца исходных данных.

Оператор составлял пакет заданий, которые в дальнейшем без его участия последовательно запускались на выполнение управляющей программой – монитором. Пакет обычно представлял собой набор перфокарт, но для ускорения работы он мог переноситься на более удобный и емкий носитель, например на магнитную ленту или магнитный диск.

Ранние системы пакетной обработки значительно сократили затраты времени на вспомогательные действия по организации вычислительного процесса, а значит, был сделан еще один шаг в направлении повышения эффективности использования компьютеров. Однако при этом программисты-пользователи лишились непосредственного доступа к компьютеру, что снижало эффективность их работы – внесение любого исправления требовало значительно больше времени, чем при интерактивной работе за пультом машины.

Еще одним важным событием стала разработка механизма виртуальной памяти. При этом способе организации вычислительного процесса программе, выполняемой под управлением ОС, предоставляется виртуальное адресное пространство оперативной памяти очень большого размера, который превосходит размер реальной физической оперативной памяти компьютера. Адреса виртуального адресного пространства динамически отображаются операционной системой на все виды памяти, которая имеется у компьютера, то есть помимо оперативной памяти на дисковую память, память на магнитной ленте и т. п. Это позволяет значительно смягчить ограничения на объем программ. Первая ОС с поддержкой виртуальной памяти была разработана в 1961 году компанией Burroughs для ее компьютера В5000. Позже эта функция стала для ОС стандартной.

Следующий важный период развития операционных систем относится к 1965-1975 годам. В это время в технической базе вычислительных машин произошел переход от отдельных полупроводниковых элементов типа транзисторов к интегральным микросхемам, что открыло путь к появлению следующего поколения компьютеров. Большие функциональные возможности интегральных схем сделали возможным реализацию на практике сложных компьютерных архитектур, таких, например, как *IBM System/360*. В этот период были реализованы практически все основные механизмы, присущие современным ОС: мультипрограммирование, мультипроцессирование, поддержка многотерминального многопользовательского режима, виртуальная память, файловые системы, разграничение доступа и сетевая работа.

Революционным событием данного этапа явилась промышленная реализация мультипрограммирования. В условиях резко возросших возможностей компьютера по обработке и хранению данных выполнение только одной программы в каждый момент времени оказалось крайне неэффективным. Решением стало мультипрограммирование – способ организации вычислительного процесса, при котором в памяти компьютера находилось одновременно несколько программ, попеременно выполняющихся на одном процессоре. Эти усовершенствования значительно повысили эффективность вычислительной системы: компьютер теперь мог использоваться почти постоянно, а не простаивать большую часть времени, как это было раньше.

Мультипрограммирование было реализовано в двух вариантах – в системах пакетной обработки и разделения времени.

Мультипрограммные системы пакетной обработки, так же, как и их однопрограммные предшественники, имели своей целью обеспечение максимальной загрузки аппаратуры компьютера, однако решали эту задачу более эффективно.

В мультипрограммном пакетном режиме процессор не простаивал, пока одна программа выполняла операцию ввода/вывода (как это происходило при последовательном выполнении программ в системах ранней пакетной обработки), а переключался на другую готовую к выполнению программу. В результате достигалась сбалансированная загрузка всех устройств компьютера, следовательно, увеличивалось число решаемых в единицу времени задач.

Однако в мультипрограммных системах пакетной обработки пользователь по-прежнему был лишен возможности интерактивно взаимодействовать со своими программами. Для того чтобы хотя бы частично вернуть пользователям ощущение непосредственного взаимодействия с компьютером, был разработан другой вариант мультипрограммных систем – системы разделения времени. Этот вариант рассчитан на многотерминальные системы, когда каждый пользователь работает за своим терминалом.

В числе первых операционных систем разделения времени, разработанных к середине 1960-х годов, были *TSS/360* (компания *IBM*), *CTSS* и *MULTICS* (*Bell Labs и General Electric*), АИСТ-0 (СССР).

Вариант мультипрограммирования, применяемый в системах разделения времени, был нацелен на создание для каждого отдельного пользователя иллюзии единоличного владения вычислительной машиной за счет периодического выделения каждой программе своей доли процессорного времени. В системах разделения времени эффективность использования оборудования ниже, чем в системах пакетной обработки, что является платой за удобства работы пользователя.

В начале 1970-х годов появились первые сетевые операционные системы, которые позволяли рассредоточить пользователей и организовать распределенное хранение и обработку данных между несколькими компьютерами, связанными электрическими связями. Любая сетевая операционная система, с одной стороны, выполняет все функции локальной операционной системы, а с другой стороны, обладает некоторыми дополнительными средствами, позволяющими ей взаимодействовать по сети с операционными системами других компьютеров.

К середине 1970-х годов наряду с мэйнфреймами широкое распространение получили миникомпьютеры, такие как *PDP-11, Nova, HP*. В миникомпьютерах впервые использовались преимущества больших интегральных схем, позволившие реализовать достаточно мощные функции при сравнительно невысокой стоимости компьютера.

Архитектура миникомпьютеров была значительно упрощена по сравнению с мэйнфреймами, что нашло отражение и в их операционных системах. Многие функции мультипрограммных многопользовательских ОС мэйнфреймов были усечены, учитывая ограниченность ресурсов миникомпьютеров. Операционные системы миникомпьютеров часто стали делать специализированными, например, только для управления в реальном времени (ОС *RT-11* для миникомпьютеров *PDP-11*) или только для поддержания режима разделения времени (RSX- 11М для тех же компьютеров). Эти операционные системы не всегда были многопользовательскими, что во многих случаях оправдывалось невысокой стоимостью компьютеров.

Важной вехой в истории миникомпьютеров и вообще в истории операционных систем явилось создание ОС *Unix*. Первоначально эта ОС предназначалась для поддержания режима разделения времени в миникомпьютере PDP-7.

С середины 1970-х годов началось массовое использование ОС *Unix*. К тому времени программный код для *Unix* был на 90 % написан на языке высокого уровня С. Широкое распространение эффективных *С*-компиляторов сделало *Unix* уникальной для того времени ОС, обладающей возможностью сравнительно легкого переноса на различные типы компьютеров: суперкомпьютеры, мэйнфреймы, миникомпьютеры, серверы и рабочие станции на базе *RISC*-процессоров, персональные компьютеры. Поскольку эта ОС поставлялась вместе с исходными кодами, то она стала первой открытой ОС, которую могли совершенствовать простые пользователи-энтузиасты. Гибкость, элегантность, мощные функциональные возможности и открытость позволили операционной системе *Unix* занять прочные позиции во всех классах компьютеров.

К наиболее важным событиям 1980-х годов можно отнести создание стека *TCP/IP*, становление Интернета, разработку новых версий ОС *Unix*, стандартизацию технологий локальных сетей, появление персональных компьютеров и операционных систем для них.

Все десятилетие было отмечено появлением новых, все более совершенных версий ОС *Unix*. Среди них были и фирменные версии *Unix*: *SunOS, HP- UX, Irix, AIX, QNX* и многие другие, в которых производители компьютеров адаптировали код ядра и системных утилит для своей аппаратуры. Разнообразие версий породило проблему их совместимости, которую периодически пытались решить различные организации. В результате были приняты стандарты *POSIX* и *XPG*, определяющие интерфейсы ОС для приложений, а специальное подразделение компании *AT&T* выпустило несколько версий *Unix System III* и *Unix System V*, призванных консолидировать разработчиков на уровне кода ядра.

В 24 января 1984 года появилась новая операционная система System 1, разработанная компанией Apple для персонального компьютера *Macintosh 128K*. Эта ОС резко отличалась ото всех существовавших тогда ОС тем, что для взаимодействия с пользователем она использовала не обычный для того времени интерфейс командной строки, а значительно более удобный графический пользовательский интерфейс, включающий столь знакомые всем нам теперь окна, меню, значки (ярлыки) файлов и программ. Это революционное изменение позволило пользователю управлять компьютером с помощью простого и удобного нового устройства – мыши, а не набирать команды на клавиатуре.

Компания *Microsoft* в своих ОС *Windows* также сделала ставку на использование оконного графического интерфейса. Изначально Windows была всего лишь графической надстройкой-программой для операционной системы 80-х и 90-х годов MS-DOS. Первая версия *Windows* появилась в 1985 году и осталась практически незамеченной, но начиная с версии 3.0, это семейство ОС оказало сильнейшее влияние на развитие индустрии персональных компьютеров. Наиболее серьезные изменения в последней версии *Microsoft Windows 8*, также произошли в области пользовательского интерфейса.

В 1987 году в результате совместных усилий *Microsoft* и *IBM* появилась первая мультипрограммная операционная система для персональных компьютеров с процессором *Intel 80286*, в полной мере использующая возможности защищенного режима – *OS/2*. Эта ОС с ее развитыми функциями многозадачности и файловой системой, снабженной встроенными средствами многопользовательской защиты, оказалась хорошей платформой для построения локальных сетей персональных компьютеров. В семействе ОС *Windows* многозадачность тоже была введена почти с первых версий. Причем долгое время существовали две реализации многозадачности. Первый вариант был характерен для тех версий *Windows*, которые фактически представляли собой надстройку (оболочку) над MS-DOS (то есть версии 1.0, 2.0, 3.0, 3.1, *Windows 95/98/МЕ*). Эта многозадачность была невытесняющей, то есть ОС не могла прервать выполняемую задачу, пока эта задача по своей инициативе не передавала управление операционной системе. Это ограничение снижало эффективность ОС. По-другому была решена проблема многозадачности в автономной линии *Windows*, начатой *Windows NT* и продолженной *Windows 2000, Windows ХР Professional, Windows Server 2003* и последующих версиях *Windows.*

В этих ОС, для которых используется обобщенное наименование «ОС семейства *Windows NT*», поддерживается эффективная вытесняющая многозадачность, полноценный многопользовательский режим, а также многопоточный режим, когда каждая задача может быть распараллелена на несколько независимо выполняющихся подзадач.

Поддержка многопользовательского и многозадачного режимов стала в современных ОС для ПК стандартом, этими свойствами обладают и две другие современных ОС для ПК стандартом, этими свойствами обладают и две другие популярные ОС, работающие на ПК, – *Mac OS X* и *Linux*.

Еще одним мощным фактором возвращения в ОС ПК многозадачности и многопользовательского режима стали локальные сети. Действительно, если компьютер подключен к локальной сети, то к его ресурсам (файлам, непосредственно подключенному принтеру и т. д.) может обращаться не только тот пользователь, который работает за клавиатурой, но и все остальные пользователи сети – если, конечно, эти ресурсы сделаны разделяемыми.

Наибольшее распространение получили сетевые оболочки *LAN Manager* компании *Microsoft* и *LAN Server* компании *IBM*, разработанные этими компаниями на основе одного базового кода.

В 1990-е годы практически все операционные системы, занимающие заметное место на рынке, стали сетевыми. Сетевые функции сегодня встраиваются в ядро ОС, являясь ее неотъемлемой частью. Операционные системы получили средства для работы со всеми основными технологиями локальных (*Ethernet, Fast Ethernet, Gigabit Ethernet, Token Ring, FDDI, ATM*) и глобальных (X.25, frame relay, ISDN, ATM) сетей, а также средства для создания составных сетей (*IP, IPX, AppleTalk, RIP, OSPF, NLSP*). В операционных системах используются инструменты мультиплексирования нескольких стеков протоколов, за счет чего компьютеры могут поддерживать одновременную сетевую работу с разнородными клиентами и серверами. Появились специализированные ОС, предназначенные исключительно для решения коммуникационных задач.

Во второй половине 1990-х годов все производители операционных систем резко усилили поддержку средств работы с Интернетом. Если до этого времени стек протоколов *TCP/IP*, на котором построен Интернет, поддерживался в основном семейством *Unix*, то теперь этот стек стал проникать во все популярные ОС. Помимо самого стека *TCP/IP*, в комплект поставки начали включать утилиты, реализующие такие популярные сервисы Интернета, как *telnet, FTP, e-mail*.

Особое внимание в течение 1990-х годов уделялось корпоративным сетевым операционным системам, дальнейшее развитие которых представляет одну из наиболее важных задач и в обозримом будущем. Данная система отличается способностью хорошо и устойчиво работать в крупных сетях, которым присуща высокая степень неоднородности программных и аппаратных средств.

**Вопрос 2. Функциональные компоненты операционной системы.**

Функции операционной системы автономного компьютера обычно группируются либо в соответствии с типами локальных ресурсов, которыми управляет ОС, либо в соответствии со специфическими задачами, применимыми ко всем ресурсам. Иногда такие группы функций называют подсистемами. Наиболее важными подсистемами управления ресурсами являются подсистемы управления процессами, памятью, файлами и внешними устройствами, а подсистемами, общими для всех ресурсов, являются подсистемы пользовательского интерфейса, защиты данных и администрирования.

**Управление процессами**. Важнейшей частью операционной системы, непосредственно влияющей на функционирование вычислительной машины, является подсистема управления процессами.

Для каждого вновь создаваемого процесса ОС генерирует системные информационные структуры, которые содержат данные о потребностях процесса в ресурсах вычислительной системы, а также о фактически выделенных ему ресурсах. Таким образом, процесс можно также определить, как некоторую заявку на потребление системных ресурсов. Чтобы процесс мог быть выполнен, операционная система должна назначить ему область оперативной памяти, в которой будут размещены коды и данные процесса, а также предоставить ему необходимое процессорное время. Кроме того, процессу может понадобиться доступ к таким ресурсам, как файлы и устройства ввода-вывода.

В мультипрограммной операционной системе одновременно может существовать несколько процессов. Часть процессов порождается по инициативе пользователей и их приложений, такие процессы обычно называют пользовательскими. Другие процессы, называемые системными, инициализируются самой операционной системой для выполнения своих функций. Поскольку процессы часто одновременно претендуют на одни и те же ресурсы, то в обязанности ОС входит поддержание очередей заявок процессов на ресурсы, например очереди к процессору, к принтеру, к последовательному порту.

Важной задачей операционной системы является защита ресурсов, выделенных данному процессу, от остальных процессов.

Одним из наиболее тщательно защищаемых ресурсов процесса являются области оперативной памяти, в которой хранятся коды и данные процесса. Совокупность всех областей оперативной памяти, выделенных операционной системой процессу, называется его адресным пространством. Говорят, что каждый процесс работает в своем адресном пространстве, имея в виду защиту адресных пространств, реализуемую ОС. Защищаются и другие типы ресурсов, такие как файлы, внешние устройства и т. д. Операционная система может не только защищать ресурсы, выделенные одному процессу, но и организовывать их совместное использование, например, разрешать доступ к некоторой области памяти нескольким процессам.

Операционная система берет на себя также функции синхронизации процессов, позволяющие процессу приостанавливать свое выполнение до наступления какого-либо события в системе, например завершения операции ввода- вывода, осуществляемой по его запросу операционной системой.

Для реализации сложных программных комплексов полезно бывает организовать их работу в виде нескольких параллельных процессов, которые периодически взаимодействуют друг с другом и обмениваются некоторыми данными. Так как операционная система защищает ресурсы процессов и не позволяет одному процессу писать в память или читать из памяти другого процесса, то для оперативного взаимодействия процессов ОС должна предоставлять особые средства, которые называют средствами межпроцессного взаимодействия.

Таким образом, подсистема управления процессами ОС выполняет следующие действия:

* распределяет процессорное время между несколькими одновременно выполняемыми в системе процессами;
* занимается созданием и уничтожением процессов;
* обеспечивает процессы необходимыми системными ресурсами;
* поддерживает синхронизацию процессов;
* реализует взаимодействие между процессами.

**Управление памятью**. Память является для процесса не менее важным ресурсом, чем процессор, так как процесс может выполняться процессором только в том случае, если его коды и данные находятся в оперативной памяти.

Управление памятью включает распределение имеющейся физической памяти между всеми существующими в системе в данный момент процессами, загрузку кодов и данных процессов в отведенные им области памяти, настройку адресно-зависимых частей кодов процесса на физические адреса выделенной области, а также защиту областей памяти каждого процесса.

Одним из наиболее популярных механизмов управления памятью в современных операционных системах является механизм так называемой виртуальной памяти. Механизм виртуальной памяти позволяет программисту писать программу так, как будто в его распоряжении имеется однородная оперативная память большого объема, часто существенно превышающего объем имеющейся физической памяти.

Основными функциями ОС по управлению памятью являются:

* отслеживание свободной и занятой памяти;
* выделение памяти процессам и освобождение памяти при завершении процессов;
* защита памяти;
* вытеснение процессов из оперативной памяти на диск, когда размеры основной памяти недостаточны для размещения в ней всех процессов, и воз- вращение их в оперативную память, когда в ней освобождается место;
* настройка адресов программы на конкретную область физической памяти.

**Управление файлами и внешними устройствами**. Способность ОС к

«экранированию» сложностей реальной аппаратуры очень ярко проявляется в одной из основных подсистем ОС – файловой системе.

При выполнении своих функций файловая система тесно взаимодействует с подсистемой управления внешними устройствами, которая по запросам файловой системы осуществляет передачу данных между дисками и оперативной памятью.

Подсистема управления внешними устройствами, называемая также подсистемой ввода-вывода, исполняет роль интерфейса ко всем устройствам, подключенным к компьютеру. Спектр этих устройств очень обширен. Номенклатура выпускаемых накопителей на жестких, гибких и оптических дисках, принтеров, сканеров, мониторов, плоттеров, модемов, сетевых адаптеров и более специальных устройств ввода-вывода, таких как, например, аналого-цифровые преобразователи, может насчитывать сотни моделей. Эти модели могут существенно отличаться набором и последовательностью команд, с помощью которых осуществляется обмен информацией с процессором и памятью компьютера, скоростью работы, кодировкой передаваемых данных, возможностью совместного использования и множеством других деталей.

Программа, управляющая конкретной моделью внешнего устройства и учитывающая все его особенности, обычно называется драйвером этого устройства (от английского *drive* – управлять, вести).

Поддержание высокоуровневого унифицированного интерфейса прикладного программирования к разнородным устройствам ввода-вывода является одной из наиболее важных задач ОС. Со времени появления ОС *Unix* такой унифицированный интерфейс в большинстве операционных систем строится на основе концепции файлового доступа. Эта концепция заключается в том, что обмен с любым внешним устройством выглядит как обмен с файлом, имеющим имя и представляющим собой неструктурированную последовательность байтов. В качестве файла может выступать, как реальный файл на диске, так и алфавитно-цифровой терминал, печатающее устройство или сетевой адаптер.

**Защита данных и администрирование**. Безопасность данных вычислительной системы обеспечивается средствами отказоустойчивости ОС, направленными на защиту от сбоев и отказов аппаратуры, ошибок программного обеспечения, а в многопользовательских ОС также средствами защиты от неcанкционированного доступа. В последнем случае ОС защищает данные от ошибочного или злонамеренного поведения пользователей системы.

**Интерфейс прикладного программирования**. Прикладные программисты используют в своих приложениях обращения к операционной системе, когда для выполнения тех или иных действий им требуется особый, привилегированный статус, которым обладает только операционная система.

Возможности операционной системы доступны прикладному программисту в виде набора функций, называющегося интерфейсом прикладного программирования *(Application Programming Interface*, *API*).

От конечного пользователя эти функции скрыты за оболочкой алфавитно- цифрового или графического пользовательского интерфейса. Для разработчиков приложений все особенности конкретной операционной системы представлены особенностями ее *API*. Поэтому операционные системы с различной внутренней организацией, но с одинаковым набором *API*-функций кажутся им одной и той же ОС, что упрощает стандартизацию операционных систем и обеспечивает переносимость приложений между внутренне различными ОС, соответствующими определенному стандарту на *API*.

Реализация системных вызовов зависит от структурной организации ОС, которая, в свою очередь, тесно связана с особенностями аппаратной платформы. Кроме того, она зависит от языка программирования. При применении языков высокого уровня функции ОС вызываются тем же способом, что и написанные пользователем подпрограммы, требуя задания определенных аргументов в определенном порядке.

**Пользовательский интерфейс**. Операционная система должна предоставлять удобный интерфейс не только для прикладных программ, но и для человека, работающего за терминалом.

Современные ОС поддерживают развитые функции пользовательского интерфейса для интерактивной работы за терминалами двух типов: алфавитно- цифровыми и графическими.

Как показал опыт использования ОС, оба типа интерфейса нужны пользователям. Массовый непрофессиональный пользователь в основном применяет графический оконный интерфейс, в то время как профессионалы задействуют оба в зависимости от решаемых задач: более мощный по функциональным возможностям интерфейс командной строки, когда нужно достичь достаточно специфического и тонкого эффекта, и графический интерфейс для рутинных операций.

**Вопрос 3. Требования к современным операционным системам.**

Главным требованием, предъявляемым к операционной системе, является выполнение ею основных функций эффективного управления ресурсами и обеспечение удобного интерфейса для пользователя и прикладных программ. Современная ОС, как правило, должна поддерживать мультипрограммную обработку, виртуальную память, свопинг, многооконный графический интерфейс пользователя, а также выполнять многие другие необходимые функции и услуги. Кроме этих требований функциональной полноты к операционным системам предъявляются не менее важные эксплуатационные требования, которые перечислены ниже.

*Расширяемость*. В то время как аппаратная часть компьютера устаревает за несколько лет, полезная жизнь операционных систем может измеряться десятилетиями. Примером может служить ОС *Unix*. Поэтому операционные системы всегда изменяются со временем эволюционно, и эти изменения более значимы, чем изменения аппаратных средств. Изменения ОС обычно заключаются в приобретении ею новых свойств, например поддержке новых типов внешних устройств или новых сетевых технологий. Если код ОС написан таким образом, что дополнения и изменения могут вноситься без нарушения целостности системы, то такую ОС называют расширяемой. Расширяемость достигается за счет модульной структуры ОС, при которой программы строятся из набора отдельных модулей, взаимодействующих только через функциональный интерфейс.

*Переносимость*. В идеале код ОС должен легко переноситься с процессора одного типа на процессор другого типа и с аппаратной платформы одного типа на аппаратную платформу другого типа. Переносимые ОС имеют несколько вариантов реализации для разных платформ, такое свойство ОС называют также многоплатформенностъю.

*Совместимость*. Существует несколько «долгоживущих» популярных операционных систем, для которых наработана широкая номенклатура приложений. Некоторые из них пользуются широкой популярностью. Поэтому для пользователя, переходящего по тем или иным причинам с одной ОС на другую, очень привлекательна возможность запуска в новой операционной системе привычного приложения. Если ОС имеет средства для выполнения прикладных программ, написанных для других операционных систем, то про нее говорят, что она обладает совместимостью с этими ОС. Следует различать совместимость на уровне двоичных кодов и совместимость на уровне исходных текстов. Понятие совместимости включает также поддержку пользовательских интерфейсов других ОС.

*Надежность* и *отказоустойчивость*. Система должна быть защищена как от внутренних, так и от внешних ошибок, сбоев и отказов. Ее действия должны быть всегда предсказуемыми, а приложения не должны иметь возможности наносить вред ОС. Надежность и отказоустойчивость ОС прежде всего определяются архитектурными решениями, положенными в ее основу, а также качеством ее реализации (отлаженностью кода). Кроме того, важно, включает ли ОС программную поддержку аппаратных средств обеспечения отказоустойчивости, таких, например, как дисковые массивы или источники бесперебойного питания.

*Безопасность*. Современная ОС должна защищать данные и другие ресурсы вычислительной системы от несанкционированного доступа. Чтобы ОС обладала свойством безопасности, она должна как минимум иметь в своем составе средства аутентификации – определения легальности пользователей, авторизации – предоставления легальным пользователям дифференцированных прав доступа к ресурсам, аудита – фиксации всех «подозрительных» для безопасности системы событий. Свойство безопасности особенно важно для сетевых ОС. В таких ОС к задаче контроля доступа добавляется задача защиты данных, передаваемых по сети.

*Производительность*. Операционная система должна обладать настолько хорошим быстродействием и временем реакции, насколько это позволяет аппаратная платформа. На производительность ОС влияет много факторов, среди которых основными являются архитектура ОС, многообразие функций, качество программирования кода, возможность исполнения ОС на высокопроизводительной платформе.

**Заключение.**

На лекции рассмотрены вопросы об общей характеристике дисциплины. Выявлено, что основой автоматизированной системы является операционная система (ОС).

ОС (англ. operating system, OS) — комплекс взаимосвязанных программ, предназначенных для управления ресурсами компьютера и организации взаимодействия с пользователем.

В логической структуре типичной вычислительной системы операционная система занимает положение между устройствами с их микроархитектурой, машинным языком и, возможно, собственными (встроенными) микропрограммами (драйверами) — с одной стороны — и прикладными программами с другой.

Разработчикам программного обеспечения операционная система позволяет абстрагироваться от деталей реализации и функционирования устройств, предоставляя минимально необходимый набор функций (см.: интерфейс программирования приложений).

С 1990-х годов наиболее распространёнными операционными системами являются системы семейства Windows, UNIX и UNIX-подобные системы.