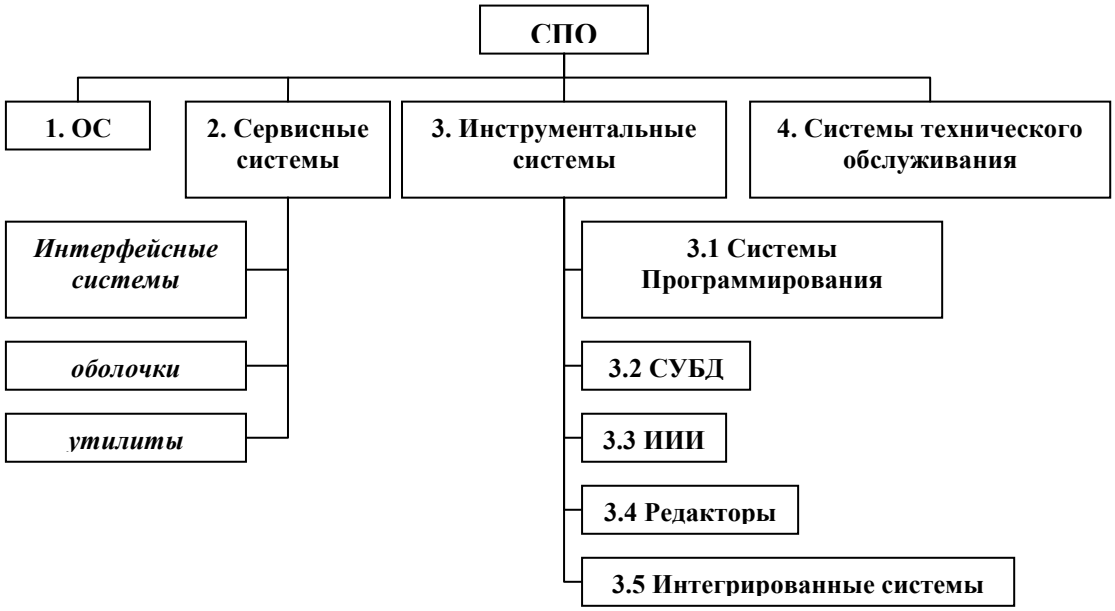
1. Раздел Общие положения

1.1 Структура системного программного обеспечения (СПО)



1. ***ОС*** – Операционная система – это комплекс программ, обеспечивающий автоматическое выполнение заданий пользователя, управление аппаратурой ***ПЭВМ***, а также планирование и эффективное использование ее ресурсов (необходимый объем *ОЗУ* и *время CPU (МП)*).

Разрядность ***МП*** определяет ***ОС***, которая может быть установлена на данном ПК.

2. Сервисные системы

Интерфейсные системы загружаются поверх ***ОС*** для улучшения пользовательского интерфейса и расширения ее возможностей (графического типа).

Оболочки – упрощают пользовательский интерфейс за счет организации системы меню с использованием функциональных клавиш.

Утилиты – служебные программы, расширяющие программный интерфейс ***ОС,*** предоставляя дополнительные функции.

3. Инструментальные системы – это совокупность программного продукта, обеспечивающего разработку информационно-программного обеспечения ПК.

3.1 Системы программирования – это язык программирования и комплекс программ для создания, отладки и выполнения нового программного продукта, а также реализация диалоговых возможностей на этапе создания.

В систему программирования входит:

1. Компилятор – программа, позволяющая проверять на наличие ошибок исходный модуль программы и переводить его в машинный код.
   Транслятор – это наиболее общее название программы-переводчика в машинные коды.
   Существует 2 типа трансляторов:
   - Наиболее распространенный тип – компилятор, после обработки получается объектный модуль программы, либо выдается сообщение об ошибке по тексту программы.

- Интерпретатор – обрабатывает программный текст построчно либо покомандно, при этом проверяет на ошибки, переводит в двоичный код и сразу выполняет команду.

2. Библиотека стандартных процедур и функций.
3. Компоновщик (linker) – редактор связи объединяет объектный модуль программы с объектными модулями процедур и функций из библиотеки.
4. Отладчик (debugger) – программа, которая выявляет логические ошибки в решении, позволяя пошагово просматривать выполнение программы, с помощью точек останова (breakpoint), а также отслеживать содержимое переменных и регистров процессора.
5. Текстовый редактор, а также сервисные программы для организации интерфейса.

Два подхода создания систем программирования:

1. автономное существование всех программных компонентов.
2. создание среды программирования.

3.2 Система управления базами данных (***СУБД***) – обеспечивает централизованное управление данными, хранящимися в базе данных – это данные, организованные специальным образом, разбитые на поля, с указанием связи этих полей между собой.

Базы данных: FoxPro, Paradox, Access, Visual Basic, Clipper, Delphi.

3.3 Инструментарий искусственного интеллекта. Направление развития: моделирование, поведение роботов, экспертные системы, решение комбинаторных задач, распознавание образов, обработка естественного языка и моделирование диалога, интеллектуальный вопрос на ответные системы.

3.4 Редакторы – это программный продукт, который служит для создания и изменения целевого документа (текстовый, графический)

3.5 Интегрированные системы – это совокупность функционально различных компонентов, способных взаимодействовать между собой путем обмена данными и объединенных единым унифицированным интерфейсом.

4. Системы технического обслуживания – предназначены для тестирования оборудования и исправления неисправностей, используется специалистами аппаратного обеспечения.

1.2 Классификация ОС. Разновидности ОС.

Классификация ОС.

1) по типу ***МП***:

8-ми разрядные ***МП*** (CPU) – CP/M, Intel 8080
16-ти разрядные ***МП*** (286 – AT) – MS-DOS, DR-DOS, Apple-DOS
32-х разрядные ***МП*** (386, 486, Pentium-I, II, III, IV) – OS/2, OS-386, MS Windows 3.11, 95, 98, NT 4.0, Windows-2000, XP, Vista, Unix, Linux.
64-х разрядные ***МП*** - Windows-7, 8,10, Linux-подобные.

2) по способу загрузки:

а) твердые – прошитые в ***ПЗУ*** (8-ми разрядные ***МП***)
б) гибридные (комбинированные) – ядро ***ОС*** прошито в ***ПЗУ***, а драйверы загружаются с магнитного диска (МД).*Драйвер* – программа, управляющая потоком информации от ПУ к центральной части и наоборот.

в) загружаемые (мягкие) – в ***ПЗУ*** находится только программа начальной загрузки ***ОС***, которая позволяет обратиться к стартовому сектору дискеты, винчестера, CD-ROM и считать программу системного загрузчика, которая «знает» все особенности загрузки программ ***ОС***, расположенных на данном диске.

3) по способу обработки заданий (по временному признаку):

- ***ОС*** реального времени (MS-DOS)
- Система с разделением времени – используется для многопользовательских систем, в которых организован мультипрограммный режим.
- Пакетный режим – все задания вначале формируются в очередь запросов, причем обработка заданий происходит без вмешательства пользователя, управлением выполнения заданий занимается оператор или администратор

4) по режиму работы:

однопрограммные *ОС* – в каждый момент времени выполняется только одна задача, второе задание входит в решение только после окончания решения первой (DOS)
мультипрограммные ОС – в процессоре все время разделяется на кванты, в решение могут быть запущены несколько задач, причем каждой из них предоставляется квант времени ***МП*** в соответствии с приоритетом в системе. Выигрыш получается за счет совмещения работы ***МП*** и обращений к внешним устройствам, т.к. ПУ могут работать без вмешательства ***МП***, которых лишь передает управляющий сигнал и данные и освобождается.

Разновидности ОС.

ОС мэйнфреймов (mainframe).

Мэйнфреймы отличаются от ПК своими возможностями ввода-вывода (терабайты данных, тысячи дисков). Представляют собой серверы для крупномасштабных электронно-коммерческих сайтов, а также серверы для транзакций в бизнесе. ***ОС*** ориентирована на множественную обработку заданий ввода-вывода – в основном пакетная обработка. Запросы, небольшие по объему операций, формируются в очередь и система должна отвечать на сотни и тысячи запросов в секунду (***ОС*** 360, ***ОС*** 390)

Серверные ОС.

Работают на серверах и одновременно обслуживают множество пользователей (юзер, user), разделяя между ними программные и аппаратные ресурсы (Unix, Windows NT-family, Linux).

Многопроцессорные ОС.

Для увеличения производительности ПК в одной системе соединяются несколько центральных процессоров. В зависимости от вида соединения ***МП*** и разделения работы системы называются параллельными компьютерами, мультикомпьютерами и многопроцессорными системами. Для них требуется специальная ***ОС***, предоставляющая собой варианты серверных ***ОС*** с расширенными возможностями связи.

<u>ОС для ПК</u> (Windows 95, 98, XP, 7,8, 10, MacOS, Linux)

I. ОС реального времени используется в системах управления производством, т.к. ПК собирают данные о промышленном процессе и используют их для управления роботами, конвейерами. Лимитировано время реакции ***ОС*** на запросы. К этой категории ***ОС*** попадают и ***ОС*** мультимедийной системы: разновидности QNX, VxWorks.

II. *Встроенные ОС* используются для Карманных ПК (КПК) и встроенных систем, имеют те же характеристики, что и системы реального времени, но ограничены мощностью (Windows CE, PalmOS).

III. *ОС для смарт-карт* (Java ориентированные) зашиты в ***ПЗУ***, имеют ограниченный набор операций и содержат интерпретатор виртуальной машины JAVA (Java Virtual Machine).

1.3 Критерии оценки ОС

При сравнительном рассмотрении различных ОС в целом или их отдельных подсистем возникает вечный вопрос – какая из них лучше и почему, какая архитектура системы предпочтительнее, какой из алгоритмов эффективнее, какая структура данных удобнее и т.п.

Очень редко можно дать однозначный ответ на подобные вопросы, если речь идет о практически используемых системах. Система или ее часть, которая хуже других систем во всех отношениях, просто не имела бы права на существование. На самом деле имеет место типичная многокритериальная задача: имеется несколько важных критериев качества, и система, опережающая прочие по одному критерию, обычно уступает по другому. Сравнительная важность критериев зависит от назначения системы и условий ее работы.

Надежность

Этот критерий вообще принято считать самым важным при оценке программного обеспечения, и в отношении ОС его действительно принимают во внимание в первую очередь.

Что понимается под надежностью ОС?

Прежде всего, ее ***живучесть***, т.е. способность сохранять хотя бы минимальную работоспособность в условиях аппаратных сбоев и программных ошибок. Высокая живучесть особенно важна для ОС компьютеров, встроенных в аппаратуру, когда вмешательство человека затруднено, а отказ компьютерной системы может иметь тяжелые последствия.

Во-вторых, способность, как минимум, диагностировать, а как максимум, компенсировать хотя бы некоторые типы аппаратных сбоев. Для этого обычно вводится избыточность хранения наиболее важных данных системы.

В-третьих, ОС не должна содержать собственных (внутренних) ошибок. Это требование редко бывает выполнимо в полном объеме (программисты давно сумели доказать своим заказчикам, что в любой большой программе всегда есть ошибки, и это в порядке вещей), однако следует хотя бы добиться, чтобы основные, часто используемые или наиболее ответственные части ОС были свободны от ошибок.

Наконец, к надежности системы следует отнести ее способность противодействовать явно неразумным действиям пользователя. Обычный пользователь должен иметь доступ только к тем возможностям системы, которые необходимы для его работы. Если же пользователь, даже действуя в рамках своих полномочий, пытается сделать что-то очень странное (например, отформатировать системный диск), то самое малое, что должна сделать ОС, это переспросить пользователя, уверен ли он в правильности своих действий.

Эффективность

Как известно, эффективность любой программы определяется двумя группами показателей, которые можно обобщенно назвать «время» и «память». При разработке системы приходится принимать много непростых решений, связанных с оптимальным балансом этих показателей.

Важнейшим показателем временно́й эффективности является ***производительность*** системы, т.е. усредненное количество полезной вычислительной работы, выполняемой в

единицу времени. С другой стороны, для диалоговых ОС не менее важно ***время реакции*** системы на действия пользователя. Эти показатели могут в некоторой степени противоречить друг другу. Например, в системах разделения времени увеличение кванта времени повышает производительность (за счет сокращения числа переключений процессов), но ухудшает время реакции.

В программировании известна аксиома: выигрыш во времени достигается за счет проигрыша в памяти, и наоборот. Это в полной мере относится к ОС, разработчикам которых постоянно приходится искать баланс между затратами времени и памяти.

Забота от эффективности долгое время стояла не первом месте при разработке программного обеспечения, и особенно ОС. К сожалению, оборотной стороной стремительного увеличения мощности компьютеров стало ослабление интереса к эффективности программ. В настоящее время эффективность является первостепенным требованием разве что в отношении систем реального времени.

Удобство

Этот критерий наиболее субъективен. Можно предложить, например, такой подход: система или ее часть удобна, если она позволяет легко и просто решать те задачи, которые встречаются наиболее часто, но в то же время содержит средства для решения широкого круга менее стандартных задач (пусть даже эти средства не столь просты). Пример: такое частое действие, как копирование файла, должно выполняться при помощи одной простой команды или легкого движения мыши; в то же время для изменения разделов диска не грех почитать руководство, поскольку это может понадобиться даже не каждый год.

Разработчики каждой ОС имеют собственные представления об удобстве, и каждая ОС имеет своих приверженцев, считающих именно ее идеалом удобства.

Масштабируемость

Довольно странный термин «масштабируемость» (scalability) означает возможность настройки системы для использования в разных вариантах, в зависимости от мощности вычислительной системы, от набора конкретных периферийных устройств, от роли, которую играет конкретный компьютер (сервер, рабочая станция или изолированный компьютер) от назначения компьютера (домашний, офисный, исследовательский и т.п.).

Гарантией масштабируемости служит продуманная модульная структура системы, позволяющая в ходе установки системы собирать и настраивать нужную конфигурацию.

Возможен и другой подход, когда под общим названием объединяются, по сути, разные системы, обеспечивающие в разумных пределах программную совместимость. Примером могут служить версии Windows NT/2000/XP, Windows 95/98 и Windows CE.

В некоторых случаях фирмы, производящие программное обеспечение, искусственно отключают в более дешевых версиях системы те возможности, которые на самом деле реализованы, но становятся доступны, только если пользователь покупает лицензию на более дорогую версию. Но это уже вопрос, связанный не с технической стороной дела, а с маркетинговой политикой.

Способность к развитию

Чтобы сложная программа имела шансы просуществовать долго, в нее изначально должны быть заложены возможности для будущего развития.

Одним из главных условий способности системы к развитию является хорошо продуманная модульная структура, в которой четко определены функции каждого модуля и его взаимосвязи с другими модулями. При этом создается возможность совершенствования отдельных модулей с минимальным риском вызвать нежелательные последствия для других частей системы.

Важным требованием к развитию ОС является ***совместимость версий снизу вверх,*** означающая возможность безболезненного перехода от старой версии к новой, без потери ранее наработанных прикладных программ и без необходимости резкой смены всех навыков пользователя. Обратная совместимость – сверху вниз – как правило, не гарантируется, поскольку в ходе развития система приобретает новые возможности, не

реализованные в старых версиях. Программа из Windows 3.1 будет нормально работать и в Windows XP; наоборот – вряд ли.

Фирмы-производители ОС прилагают максимум усилий для обеспечения совместимости снизу вверх, чтобы не отпугнуть пользователей. Но при этом фирмы стараются в каждую новую версию заложить какую-нибудь новую конфетку, которая побудила бы пользователей как можно скорее купить ее.

Совместимость версий – благо для пользователя, однако на практике она часто приводит к консервации давно отживших свой век особенностей или же просто неудачных решений, принятых в ранней версии системы. В документации подобные архаизмы помечаются как «устаревшие» (obsolete), но полного отказа от них, как правило, не происходит (а вдруг где-то еще работает прикладная программа, написанная двадцать лет назад с использованием именно этих средств?).

Как правило, наиболее консервативной стороной любой ОС являются не алгоритмы, а структуры системных данных, поэтому дальновидные разработчики заранее строят структуры «на вырост»: закладывают в них резервные поля, используют переменные вместо некоторых констант, устанавливают количественные ограничения с большим запасом и т.п.

Мобильность

Под ***мобильностью*** (portability) понимается возможность переноса программы (в данном случае ОС) на другую аппаратную платформу, т.е. на другой тип процессора и другую архитектуру компьютера. Здесь имеется в виду перенос с умеренными трудозатратами, не требующий полной переработки системы.

Свойство мобильности не столь однозначно положительно, как может показаться. Чтобы программа была мобильна, при ее разработке следует отказаться от глубокого использования особенностей конкретной архитектуры (таких, как количество и функциональные возможности регистров процессора, нестандартные команды и т.п.). Мобильная программа должна быть написана на языке достаточно высокого уровня (часто используется язык С), который можно реализовать на компьютерах любой архитектуры. Платой за мобильность всегда является некоторая потеря эффективности, поэтому немобильные системы распространены достаточно широко.

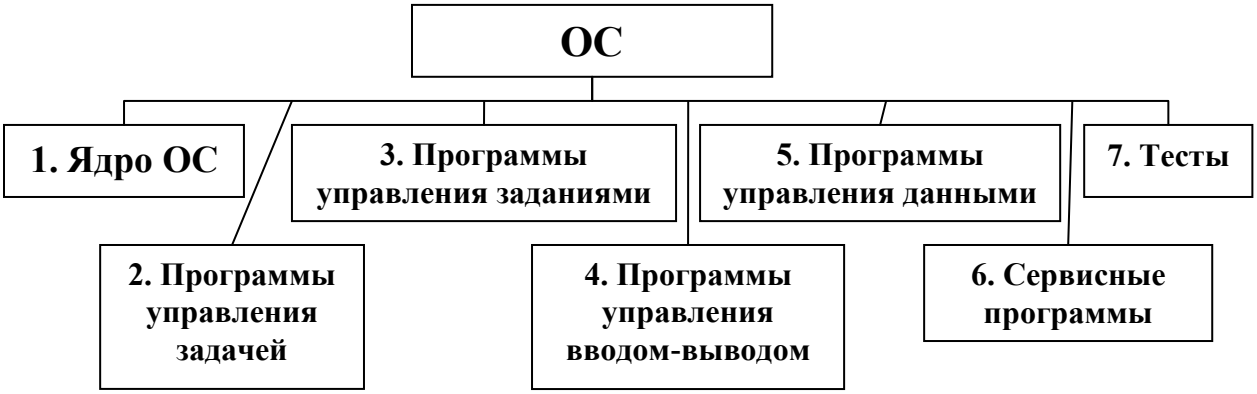
С другой стороны, история системного программирования усеяна останками замечательных, эффективных и удобных, но немобильных ОС, которые вымерли вместе с процессорами, для которых они предназначались. В то же время мобильная система UNIX продолжает процветать четвертый десяток лет, намного пережив те компьютеры, для которых она первоначально создавалась. Примерно 5-10% исходных текстов UNIX написаны на языка ассемблера и должны переписываться заново при переносе на новую архитектуру. Остальная часть системы написана на С и практически не требует изменений при переносе.

Некоторым компромиссом являются ***многоплатформенные*** ОС (например, Windows NT), изначально спроектированные для использования на нескольких аппаратных платформах, но не гарантирующие возможность переноса на новые, не предусмотренные заранее архитектуры.

1.4 Функции любой ОС.

1) выделение ресурсов и управление ими (время ***МП*** и объем ***ОЗУ*** – аппаратные ресурсы) и программные ресурсы (драйверы и необходимые системные обработчики)
2) загрузка программ в ОП
3) пересылка информации в устройствах связи
4) управление операциями ввода-вывода (драйверы)
5) обслуживание системы прерываний. ***Прерывание*** – это ситуация, вызывающая приостановку выполнения текущей команды ***МП*** и переключение на системную обработку данной ситуации
6) управление и поддержка файловой системы
7) конфигурирование технических средств – настройка ***ОС*** на конкретные устройства, которые могут быть использованы самой системой и пользователем (user)
8) Обработка командного языка
9) Защита и учет использования ресурсов
10) Диспетчеризация процессов вычислительной системы
11) Распределение памяти внешних Запоминающих Устройств (ЗУ)

1.5 Структура ОС



1. Ядро (супервизор, режим ядра – это особый режим при выполнении команд ***МП***, который позволяет напрямую обращаться к устройствам ПЭВМ и в любой раздел ОП) включает в себя драйверы стандартных ПУ и обработчики системных вызовов.
   Часто зашито в ***ПЗУ*** (Basic Input-Output System – BIOS – базовая система ввода-вывода).

2. Программы управления задачей – обеспечивают загрузку программных кодов в ***ОЗУ***, запуск на выполнение, выделение раздела ***ОЗУ***, а так же очищение раздела памяти из ***ОЗУ*** (выгрузка) после выполнения программы, либо оставляет программу резидентной в ***ОЗУ***.

3. Программы управления заданиями – обеспечивают мультипрограммный режим с использованием программы-планировщика заданий, распределяет разделы памяти (***ОЗУ***) между задачами, обеспечивая защищенность и безопасность работы каждой из них.

4. Программы управления вводом-выводом – обеспечивают операции ввода-вывода (за счет драйверов), обработку очередей запросов на ввод-вывод и распределение этих запросов между устройствами и наборами данных.

5. Программы управления данными – обеспечивают методы доступа к данным библиотекам, каталогам (связаны со средствами управления задачей), а также передают в ***ОЗУ*** и к устройствам ввода-вывода.

6. Сервисные программы – расширяют возможности ***ОС*** (утилиты) и пользовательского интерфейса.

7. Тесты – для диагностики и тестирования устройств и программ.