**Лекция 4 Микро ядерная структура (модель клиент-сервер и модель ОС на базе микроядра)**

1. Модель клиент-сервер

Модель клиент-сервер — это еще один подход к структурированию ОС. В широком смысле такая модель предполагает наличие программного компонента — потребителя какого-либо сервиса (клиента) и программного компонента — поставщика этого сервиса (сервера). Взаимодействие между клиентом и сервером стан- партизуется, так что сервер может обслуживать клиентов, реализованных различными способами и, может быть, разными производителями. При этом главным является требование, чтобы клиенты запрашивали услуги сервера понятным ему способом. Инициатором обмена обычно является клиент, который посылает запрос на обслуживание серверу, находящемуся в состоянии ожидания запроса (рис.1). Один и тот же программный компонент может быть клиентом по отношению к одному виду услуг и сервером для другого вида услуг. Модель клиент-сервер является скорее удобным средством ясного представления функций того или иного программного элемента в той или иной ситуации, нежели технологией. Эта модель успешно применяется не только при построении ОС, но и на всех уровнях программного обеспечения и имеет в некоторых случаях более узкий, специфический смысл, сохраняя, естественно, при этом все свои основные черты

В целях обеспечения эффективности и целостности работы – это реализуется исходя из двух режимов: **режима пользователя** (uspf mode) и **режима ядра** (kernel mode).

Операционная система разбивается на несколько подсистем, каждая из которых выполняет определенный набор сервисных функций, например управление памятью, создание или планирование процессов. Каждая подсистема реализуется в пользовательском режиме. Клиент, которым может быть либо другой компонент ОС, либо прикладная программа, запрашивает сервис, посылая сообщение на сервер. Ядро ОС, работая в привилегированном режиме, доставляет сообщение нужному серверу, сервер выполняет операцию, после чего ядро возвращает результаты клиенту с помощью другого сообщения.

2. Режим пользователя

Это менее привилегированный режим работы процессора по сравнению с режимом ядра. Он не предусматривает прямого доступа к аппаратуре. Выполняющийся в этом режиме код непосредственно имеет дело лишь с объектами своего адресного пространства (рис. 9.5). Системные службы он вызывает через интерфейсы прикладных программ (API — Application Program Interface). Поддерживающие их приложения и подсистемы работают в режиме пользователя. При запуске приложения создается *процесс* (process), реализованный в виде *объекта* (object). Объект состоит из исполняемой программы, пространства адресов виртуальной памяти и одного или нескольких потоков.

Особенности процесса режима пользователя:

 не имеет прямого доступа к оборудованию. Это сделано в целях защиты от неверно работающих приложений или от несанкционированного доступа. Запросы на использование аппаратных ресурсов должны быть разрешены компонентом режима ядра;

ограничен размерами выделенного адресного пространства. Ограничение размера памяти, используемой процессом, позволяет обеспечить дополнительную защиту ОС. Это ограничение ус­танавливается путем выделения процессу диапазона фиксирован­ных адресов;

может быть выгружен из физической памяти в *виртуальную память* (VRAM — virtual memory) на жестком диске. Виртуальная память использует пространство жесткого диска как дополнительную оперативную память. В результате процесс режима пользователя получает доступ к памяти, размер которой превышает объем ОЗУ;

приоритет процесса данного типа ниже, чем у процессов режима ядра, поэтому в сравнении с последними ему, как правило, предоставляется меньше процессорного времени. Это предохраняет ОС от снижения производительности или возникновения задержек, связанных с ожиданием завершения работы приложений.



Рис. Представление режима пользователя