*Московский Технический Университет*

*Связи и Информатики*

Кафедра теории вероятностей и прикладной математики

**Операционные системы и сети**

Реферат

**«**Архитектура операционной системы Windows»

Выполнила

Студентка

группы БПМ1301

Голяева А.В.

Проверила

Таташев А.Г.

Москва 2016

 **Введение**

Компьютеры играют очень важную роль в нашей современной жизни. Они проникли во все сферы деятельности человека. Они позволяют хранить огромные объёмы информации и моментально получать к ним доступ; они способны также обрабатывать эту информацию во много раз быстрее человека, работают дольше по времени и с меньшим количеством ошибок. Если раньше на обработку какой-либо информации человеком уходили дни, то сейчас с помощью компьютера, это занимает секунды, что позволяет сконцентрироваться на задаче, а не на вычислениях. Компьютеры применяются в медицине, что является очень полезным фактом; являются самым дешёвым и быстрым средством общения; при правильном использовании служат нам незаменимым помощником в учёбе, а также просто для отдыха.

Домашний компьютер сейчас есть у большинства людей. И он может оказать нам неоценимую пользу. Он является источником новой информации. Вряд ли мы можем представить свою жизнь без него. Он имеет множество преимуществ: мы можем на нём учиться, писать, читать, смотреть фильмы, играть в игры, общаться с людьми, которые находятся за сотни километров от нас, создавать программы и проекты, рисовать.

Для начала взглянем на минимальный состав компьютера, в который входит системный блок (который и позволяет ему работать), монитор ( чтобы видеть всё то, с чем мы работаем) и клавиатура (чтобы вводить данные). Также для работы с компьютером используются компьютерная мышь, колонки, принтер, сканер, микрофон, камера.

В системном блоке находятся материнская плата, видеокарта, винчестер( жесткий диск), процессор, блок питания, оперативная память, системы охлаждения, дисковод.
 **Что такое операционная система?**

«Операционная система, сокр. ОС (англ. operating system, OS) — комплекс управляющих и обрабатывающих программ, которые, с одной стороны, выступают как интерфейс между устройствами вычислительной системы и прикладными программами, а с другой стороны — предназначены для управления устройствами, управления вычислительными процессами, эффективного распределения вычислительных ресурсов между вычислительными процессами и организации надёжных вычислений. Это определение применимо к большинству современных операционных систем общего назначения».

То есть операционную системы можно назвать движущей силой нашего компьютера. Именно она даёт нам возможность взаимодействовать с компьютером. Она делает информацию, которую выдаёт нам компьютер, понятной для нашего восприятия, и в то же время переводит то, что мы вводим на компьютерный язык, чтобы компьютер смог выполнить поставленную перед ним задачу. Она является связующим звеном между аппаратурой компьютера и человеком.

**Функции операционной системы:**

 1)осуществление диалога с пользователем;

 2)ввод, вывод и управление данными;

3)планирование и организация процесса обработки программ;

4)распределение ресурсов (оперативной памяти и кэша, процессора, внешних устройств); 5)запуск программ на выполнение;

6)передача информации между различными внутренними устройствами;

7)программная поддержка работы периферийных устройств (дисплея, клавиатуры, дисковых накопителей, принтера и др.).[

Благодаря операционной системе облегчается работа человека с компьютером, так как ОС скрывает сложные и ненужные подробности взаимодействия с аппаратурой.

**Архитектура ОС Windows**

Архитектура ОС Windows имеет модульную структуру. Из этого можно сделать вывод, что она состоит из разделяемых компонентов, любой из которых жестко отвечает за собственные функции. Она состоит из двух уровней: компонентов, которые работают в режиме пользователя (user mode), и компонентов, работающих в режиме ядра (kernel mode). Это очень удобно, потому что пользователь, имея доступ к системным программам и внутренним компонентам ОС, может случайно что-то изменить или удалить. Это спровоцирует поломку системы. Поэтому с компьютером мы работаем через пользовательский режим и его приложения, а в режиме ядра выполняются наши команды.

У пользовательского режима и режима ядра есть свои привилегии. У режима ядра их больше. В операционной системе они разделены на уровни

**Режим ядра**

Сначала мы рассмотрим режим ядра, который, непременно, является очень важной и неотъемлемой частью операционной системы.

Ядро — центральная часть операционной системы, управляющая выполнением процессов, ресурсами вычислительной системы и предоставляющая процессам координированный доступ к этим ресурсам. [3]

По сути это некоторая абстракция, которая позволяет получить доступ к ресурсам вычислительной системы, который нужны пользователю для работы с компьютером. Работа ядра полностью скрыта от пользователя, и он не имеет к нему доступа. Существует несколько типов архитектур ядер операционных систем:

 1) Монолитное ядро;

 2)Микроядро;

3) Экзоядро;

4)Наноядро;

5)Гибридное ядро.

« Монолитное ядро — классическая архитектура ядер операционных систем. Монолитные ядра предоставляют богатый набор абстракций оборудования.

Все части монолитного ядра работают в одном адресном пространстве».[4] То есть в памяти нашего компьютера хранится информация, и каждой её составляющей соответствует число – адрес, который определяет место её хранения. Кстати, монолитное ядро сейчас всё равно является самой распространённой архитектурой ядер операционных систем.

 Монолитные ядра долго развивались и усовершенствовались. Сейчас они являются наиболее архитектурно зрелыми и пригодными к эксплуатации. Но, к сожалению, монолитное ядро имеет свои недостатки. Оно работает в одном адресном пространстве. Если вдруг произойдёт сбой в одном из компонентов, то нарушится работоспособность всей системы. Эти сбои невозможны в модульном ядре, о котором мы сейчас поговорим.

«Модульное ядро — современная, усовершенствованная модификация архитектуры монолитных ядер операционных систем компьютеров. В отличие от «классических» монолитных ядер, считающихся ныне устаревшими, модульные ядра, как правило, не требуют полной перекомпиляции ядра при изменении состава аппаратного компьютера». [5] Взамен этим функциям модульные ядра просто дополнительно загружают модули ядра, которые в свою очередь поддерживают то или иное аппаратное обеспечение. Например, модульное ядро может загрузить нужный для работы драйвер.

Модульное ядро может загрузить модуль прямо во время работы системы, не перезагружая её. А также подгрузка модуля может выполняться при перезагрузке ОС. В это время ОС перестраивается на загрузку определенных модулей. Но, не смотря на новые возможности, модульные ядра всё равно остаются монолитными.

 Они по-прежнему работают в адресном пространстве. Это даёт им возможность использовать все функции ядра. «Модульные ядра удобнее для разработки, чем традиционные монолитные ядра, не поддерживающие динамическую загрузку модулей, так как от разработчика не требуется многократная полная перекомпиляция ядра при работе над какой-либо его подсистемой или драйвером. Выявление, локализация, отладка и устранение ошибок при тестировании также облегчаются».[6] То есть если нужно усовершенствовать функции модульного ядра, то сначала можно улучшить одну из его составляющих частей, например, какой-нибудь драйвер, а потом загрузить его к ядру. Эта задача гораздо усложняется в случае с монолитным ядром. Его придётся полностью переделывать, чтобы поместить туда новый драйвер.

Функции, которые входят в состав ядра, решают очень важные задачи. Они переключают контексты, занимаются организацией процесса вычислений, загружают и выгружают страницы, обрабатывают прерывания.

А в свою очередь приложения, работающие из режима пользователя, могут обращаться к ядру, чтобы выполнить нужные действия. Скорость выполнения функция ядра является показателем производительности ОС. Все модули ядра или большая их часть находятся в ОС. Ядро является главной составляющей компьютерной системы, благодаря которой она работает. По этой причине так необходимо, чтобы коды ядра были надежны.

 Ядро состоит из следующих слоёв:

1)Средства аппаратной поддержки;

2)Машинно-зависимые компоненты;

 3)Базовые механизмы ядра;

4)Менеджеры ресурсов;

 5)Интерфейс системных вызовов.

 Программное обеспечение, находящееся в режиме ядра: ·

может иметь прямой и неограниченный доступ к аппаратному обеспечению; ·

 имеет доступ ко всей памяти компьютера;

· не может быть вытеснено в страничный файл на жестком диске; ·

 имеют больший приоритет по сравнению с процессами пользовательского режима

Сама архитектура ядра позволяет защитить его компоненты от изменения их другими программами.

 Драйвер – это компьютерные программы, благодаря которым ОС может работать с устройствами компьютера. Драйвер обрабатывает наши запросы, которые мы вводим в режиме пользователя. Затем он переводит их на машинный язык (с высокого уровня на низкоуровневый), который понятен компьютеру, тем самым обеспечивая нам полное взаимопонимание.

Основная часть кода Windows выполняется в режиме ядра. Если драйвер будет содержать вирус, то он может сломать ОС. Когда мы загружаем драйверы из интернета, то нужно быть осторожными, так как они сразу переходят в режим ядра и получают полный доступ ко всем данным операционной системы.

Поэтому очень важно, чтобы компоненты, которые работают в этом режиме, были хорошо проверены и протестированы, не содержали ошибок.

 А теперь рассмотрим режим пользователя, который является нам гораздо ближе. Ведь работать с ядром напрямую мы не можем, мы можем лишь отдавать команды через режим пользователя.

**Режим пользователя**

 Режим пользователя имеет гораздо меньше привилегий по сравнению с режимом ядра. У него нет прямого доступа к оборудованию. А также он имеет ограниченный доступ к памяти.

«Режим пользователя состоит из подсистем, которые передают запросы ввода и вывода соответствующему драйверу режима ядра посредством менеджера Ввода-вывода. Уровень пользователя состоит из двух подсистем — подсистема окружения(Environment) и интегральная подсистема (Integral)». [

Подсистема окружения нужна для того, чтобы запускать приложения, которые написаны для разных видов ОС. Обратим внимание на то, что ни у какой из подсистем окружения нет прямого доступа к аппаратной части компьютера.

 У ОС есть менеджер Вирт. Памяти. Этот менеджер имеет привилегии, так как работает в режиме ядра. Именно он обеспечивает доступ к ресурсам памяти. В состав системы окружения входят следующие подсистемы: Win32, POSIX, OS/2. Самой важной из них является Win 32. Она предоставляет компьютерным программам интерфейс программирования приложения (англ. application programming interface, API). Две другие подсистемы тоже предоставляют свой интерфейс программирования приложения. Но для того чтобы им получить пользовательский ввод и показать результаты, им нужно использовать подсистему Win32.

 Без подсистемы Win32 Windows работать не может, поэтому Win32 работает всегда. Это её главное отличие от двух других подсистем. Эта подсистема обрабатывает все, что связано с клавиатурой, мышью и экраном, и нужна даже на серверах в отсутствие интерактивных пользователей. Также Win32 управляет окнами и базовыми сервисами. «Подсистема окружения OS/2 поддерживает неграфические 16-разрядные приложения операционной системы OS/2 и эмулирует систему OS/2 2.1.x.

 Подсистема окружения POSIX поддерживает приложения, написанные в соответствии со стандартом POSIX.1»[9] Интегрированная подсистема работает от имени подсистемы окружения. Она наблюдает за несколькими функциями операционной системы. В её состав входят подсистемы безопасности, службы рабочей станции и службы сервера. «Служба безопасности обращается с маркерами доступа, позволяет или запрещает доступ к учетной записи пользователя, обрабатывает запросы авторизации и инициирует процесс входа пользователя в систему.

 Служба Рабочая станция обеспечивает доступ компьютера к сети — является API для сетевого редиректора. Служба Сервер позволяет компьютеру предоставлять сетевые сервисы».Теперь можно рассмотреть более полную схему архитектуры ОС.

**Главные отличия двух режимов**

Чем же отличаются программы, работающие в режиме пользователя от программ, работающих в режиме ядра? Отличия процессов, работающих в режиме пользователя, от процессов, работающих в режиме ядра:

не имеют прямого доступа к оборудованию, все запросы на использование аппаратных ресурсов должны быть разрешены компонентом режима ядра;

ограничены размерами выделенного адресного пространства;

 могут быть выгружены из физической памяти в виртуальную на жестком диске;

 приоритет процессов данного типа ниже приоритета процессов режима ядра, это предохраняет ОС от снижения производительности или задержек, происходящих по вине приложений.

**Пример работы ОС**

 А сейчас мы посмотрим, как работает ОС. Допустим на надо написать и распечатать какой-нибудь текст. Что мы делаем? Открываем текстовый редактор, набираем текст, сохраняем этот документ, а потом распечатываем.

А что же делает наша операционная система, и как взаимодействуют её компоненты? Во-первых, клавиатура и мышь отправляют введённые нами данные в ОС. Затем ОС решает, что текстовый редактор является активной программой и использует нужные модули для работы с ней. ОС принимает данные, которые мы ввели, и отправляет их к себе на хранение. Дело в том, что ОС обладает небольшим объёмом и хранит информацию временно, только при работе с ней. Если вдруг выключить компьютер, то вся информация, хранящаяся в ОС, не сохранится. Поэтому всю информацию мы сохраняем на жёстком диске. Каждое наше редактирование из текстового редактора посылается ОС в центральный процессор. При этом команды переводятся на машинный язык. Процессор работает уже в режиме ядра. На протяжении этого времени ОС предоставляет нам всю информацию на мониторе нашего компьютера с помощью графической карты. Теперь нам нужно сохранить текст. Мы нажимаем кнопку Сохранить. Тем временем текстовый редактор отправляет запрос в ОС. ОС отвечает, позволяя нам выбрать место сохранения файла и его имя. После того как мы выберем нужные нам параметры, ОС направляет эти данные на хранение в устройство с постоянной памятью. Теперь текст нужно распечатать. Нажимаем кнопку Печать. Теперь ОС принимает наши данные и отправляет их на принтер до нужного порта, который мы указываем.