Операционные системы

Курс лекций для гр. 4057/2

Лекция №13

Вопросы к лекции 12

1. Понятно, как экспериментально измеряется MTBF порядка часов или дней: несколько образцов ставятся на испытания на месяцы или годы, и статистика их отказов усредняется по времени. А как оценить MTBF очень надежного элемента (например, интегральной схемы), если оно порядка сотен лет?
2. Почему надежность (безотказность) вычислительной аппаратуры удалось повысить в последние десятилетия на несколько порядков, а сделать это для программ – нет?
3. Одна из причин ненадежности программных продуктов (ПП) – их б**о**льшая сложность по сравнению с аппаратурой. Какое простое рассуждение подтверждает, что сложность нетривиальных программ действительно выше, чем сложность компьютеров, на которых они выполняются ?
4. Предположим, ферма содержит 10 тысяч серверов повышенной надежности с MTBF=30 лет. Сколько в среднем серверов будут отказывать за день после начального периода «выгорания слабых элементов»?
5. В протоколах TCP/IP реализовано циклическое кодирование 64-байтных блоков передаваемых данных (с восстановлением 2-кратных и обнаружением 3-х кратных ошибок при декодировании) и с переспросом непринятых или сильно искаженных блоков. Каким видам резервирования это соответствует ?
6. Какой из четырех видов резервирования пригоден для восстановления после сбоев, но бесполезен при устойчивых отказах ?
7. Что должна делать система, если в схеме дублирования возник сигнал ошибки ?
8. Предложите схему и принцип четырехкратного модульного резервирования.
9. Почему это необходимо делать? Что случится, если не выполнять redo?

Содержание

Назначение защиты (1)

Защита (protection)–это предотвращение случайных или умышленных нарушений вычислительного процесса

- Защита противодействует несанкционированному чтению /изменению / уничтожению информации:
  - случайному из-за программных ошибок или аппаратных отказов
  - преднамеренному - в результате *атаки* на вычислительную систему (нарушитель или вирус)
- Программно-аппаратные механизмы защиты в ОС управляют доступом процессов к ресурсам вычислительной системы

Близкое понятие *безопасность (security)* – более широкое, охватывает весь комплекс мер защиты, включая:

- организационные - работу сисадмина, выполняющего *политику безопасности* по настройке средств защиты для определенного круга пользователей)
- средства *засекречивания* (шифрования) информации

Назначение защиты (2)

- *Объекты защиты* – ресурсы вычислительной системы: файлы, процессы, память, устройства вв/вы, – в конечном счете данные: либо сама защищаемая информация, либо коды процессов, либо служебные структуры данных ОС (напр., дескрипторы)
- *Субъекты защиты* – процессы и пользователи, т.е. в конечном счете процессы, поскольку пользователь представлен в ОС процессами, ему принадлежащими
- *Доступ субъекта к объекту* – выполнение операций с данными: чтение, запись, удаление, выполнение кода
- Назначение защиты – управление доступом субъектов к объектам

Основные функции подсистемы защиты

1. Описание защиты: определение защищаемых объектов
2. Раздача прав доступа (разрешений на операции доступа) субъектам защиты
3. Контроль полномочности доступа – статический и динамический

Формальная модель защиты

- описывает отношения разрешенного доступа между субъектами и объектами

- $Y = \{y_1, y_2, ..., y_n\}$ – множество объектов защиты
- $S = \{s_1, s_2, ..., s_m\}$ – множество субъектов защиты
- $P = \{p_1, s_2, ... s_r\}$ – множество операций доступа
  - напр., P = {R, W, E} для файлов
- $A = \{a_1, a_2, ..., a_k\}$ – множество прав доступа; $a_i \subseteq P$ (i=1, ..., k); $k \le 2^r$
- $X = \{x_1, x_2, ..., x_t\}$ – множество классов привилегий (полномочий)

Модель защиты состоит из двух отображений:

$Z : S \rightarrow X$ - определяет, к каким классам относятся субъекты

$F : X \times Y \rightarrow A$ (иначе, $A = f(X, Y)$) - определяет *состояние* системы защиты

(Вопрос 1)

Матрица доступа

- представление функции f (X, Y) двухвходовой таблицей

Классы приви-легий	Объекты			
	y_1	y_2	…	y_n
x_1	a^{11}	a^{12}	…	a^{1n}
x_2	a^{21}	a^{22}	…	a^{2n}
…	…	…	…	…
x_t	a^{t1}	a^{t2}	…	a^{tn}

Эта таблица не существует как целое

Для каждого типа объектов создается отдельная подматрица, и существует три способа ее хранения:

- по столбцам
- по строкам
- по элементам матрицы

Хранение матрицы доступа по столбцам

Элементы одного столбца хранятся в дескрипторе объекта в виде *списка доступа* к объекту - упорядоченного набора пар

<класс_привилегий, набор_прав>

Пример – защита файлов в Unix:

Классы привилегий	Файлы			
	y_1	y_2	…	y_n
Владелец	R -X	RWX	…	RWX
Групповой пользователь	R - X	- - X	…	- - X
Общий пользователь	R - X	- - -	…	- - X
Супер-пользователь	RWX	RWX	…	RWX

В Unix - столбец фиксированной длины – *вектор доступа*

В Windows - список переменной длины – более гибкий способ

(Вопросы 2, 3)

Хранение матрицы доступа по строкам

Элементы одной строки образуют *список прав* у класса привилегий

- Пример: таблица страниц процесса – это список виртуальных страниц, доступных процессу (остальные просто не видны)
- Плюс код защиты в дескрипторе каждой страницы определяет разрешенные операции, т.е. права доступа:

	Z	Защита	**R**	**M**	**A**	№ кадра памяти

- В архитектуре Intel код защиты – двухбитовый: R, RW, E
- В режиме ядра программам ОС разрешен доступ ко всей памяти, так что здесь мы имеем N+1 классов привилегий, где N – число активных процессов.

Сравнение способов хранения матрицы доступа

- Список доступа хорошо соответствует нуждам пользователей: при создании нового объекта его владелец легко может сразу определять список доступа, а в дальнейшем его изменять

  ✓ Минус: информация о защите для конкретного класса привилегий не локализована, что затрудняет ее поиск и использование (ведь права проверяются при каждой операции доступа)

- Для списка прав – все наоборот

- Большинство ОС используют комбинацию 1 и 2 способов: при первой попытке доступа к объекту проверяется список доступа и если доступ разрешен, создается список прав, присоединяемый к процессу

  ✓ Напр., такой список хранится в таблице открытых файлов процесса

Правила разграничения доступа (1)

A. Разграничение по усмотрению пользователя (Discretionary access control):

- Владелец объекта может произвольно ограничивать доступ других субъектов
- Существует хотя бы один привилегированный пользователь (администратор), имеющий возможность обратиться к любому объекту с любой возможной операцией доступа
- При создании объекта его владельцем назначается тот, кто его создал. В дальнейшем субъект, обладающий необходимыми правами, может назначить объекту нового владельца – себя. (Вопрос 4)
- Владелец обычно определяет список доступа для создаваемого объекта; по умолчанию новый объект наследует атрибуты защиты от родительского объекта (процесса, каталога, контейнера и т.д.).

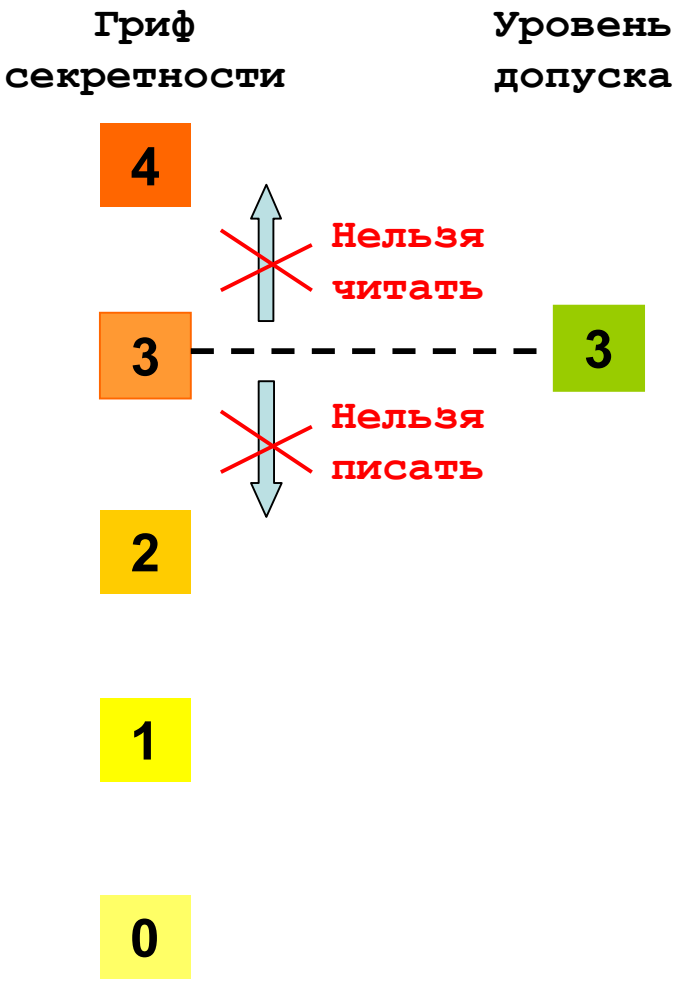
Правила разграничения доступа (2)

Б. Полномочное, или мандатное (mandatory) разграничение доступа

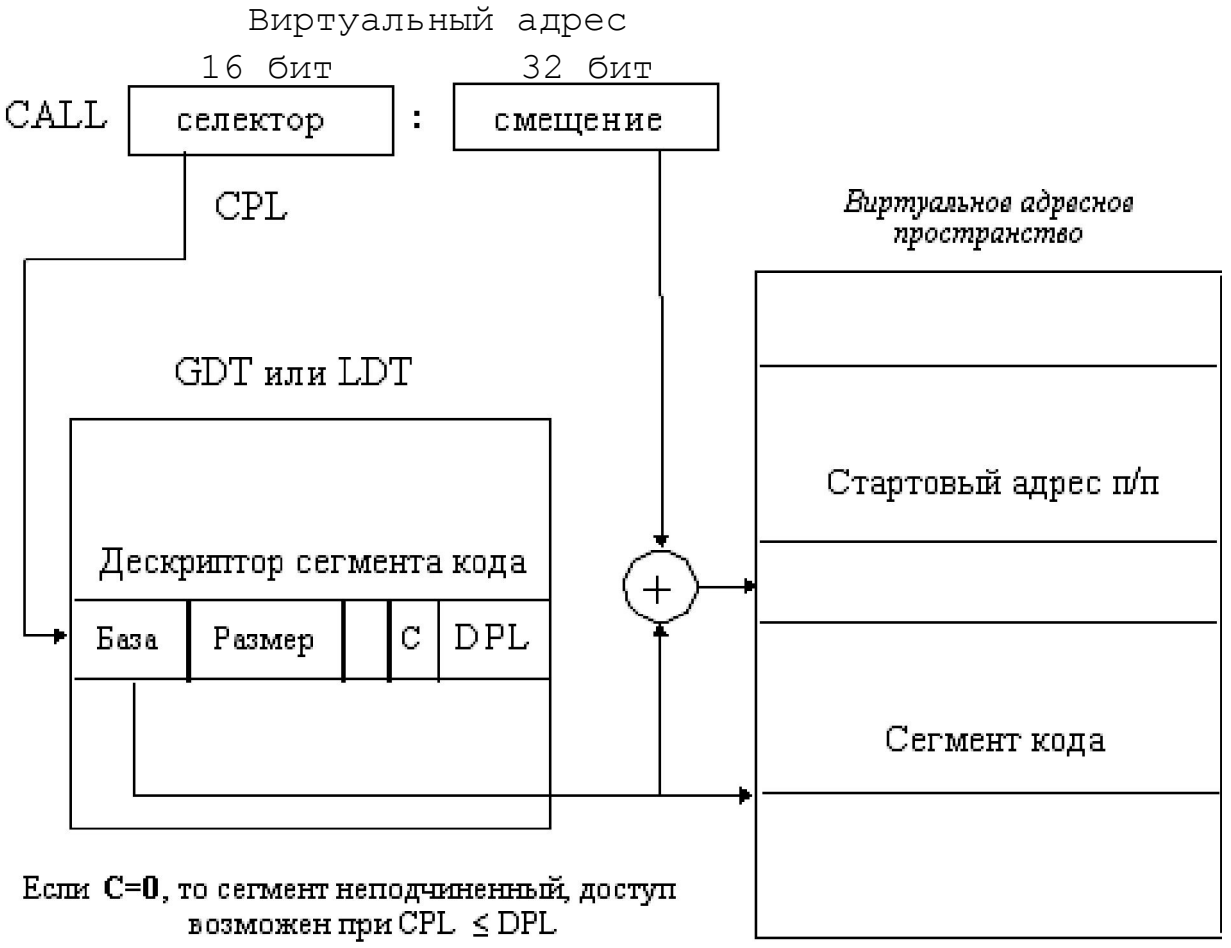
К правилам варианта А добавляются следующие:

- Каждый объект имеет *гриф секретности* – числовой эквивалент степени защищенности. К объектам с нулевым грифом, т.е. несекретным, администратор имеет полный доступ
- Каждый субъект имеет *уровень допуска.* Чем выше его значение, тем больший допуск
- Если гриф секретности объекта выше уровня допуска субъекта, то для него доступ к объекту по чтению запрещен независимо от состояния матрицы защиты. Это *правило NRU* (Not Read Up – не читать выше)
- Если гриф секретности объекта ниже уровня допуска субъекта, то для него доступ к объекту по записи запрещен независимо от состояния матрицы защиты. Это *правило NWD* (Not Write Down – не писать ниже)
- Понизить гриф секретности объекта может только субъект, обладающий специальной привилегией

Схема мандатного разграничения доступа



Аппаратная поддержка защиты сегментов памяти в процессорах Intel (1)



CPL - current privilege level

DPL - descriptor privilege level

Схема вызова подпрограммы из другого сегмента памяти

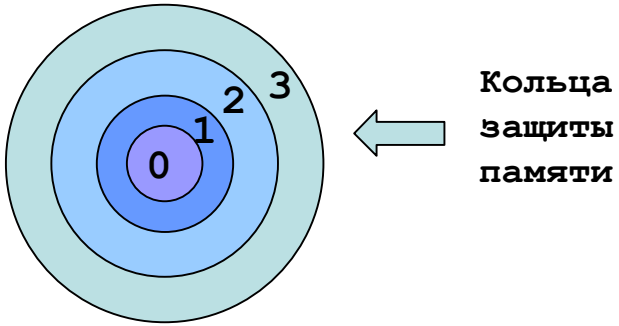
Аппаратная поддержка защиты сегментов памяти в процессорах Intel (2)

Виртуальный адрес складывается из 16-битного *селектора* сегмента и 32-битного смещения

При его загрузке ОС находит в таблице сегментов соответствующий *дескриптор* сегмента и загружает его в соответствующий *дескрипторный* регистр (один из 6 – невидимых из программы)

Уровень привилегий сегмента (privilege level, PL) – числовой идентификатор 0..3 – в порядке уменьшения привилегий

Значение CPL сравнивается с DPL при каждом обращении к памяти, и при CPL > DPL генерируется прерывание по защите памяти, т.е. реализуется правило NRU мандатного разграничения доступа.



Аппаратная поддержка защиты сегментов памяти в процессорах Intel (3)

- Итак, в архитектуре Intel386 есть возможность организации четырех вложенных уровней привилегий процессов и данных – т.н. *колец защиты*
- Процесс, выполняющийся в N-ом кольце, не может обращаться к сегментам в кольцах с номерами < N, а может только с номерами ≥ N
- Обычно приложения выполняются в 3 кольце защиты, а ядро ОС – в нулевом
- 1 и 2 кольца предназначены для системных утилит и промежуточного ПО (напр., СУБД)
  - ✓ используются редко: напр., в OS/2
  - ✓ в Windows – не используются (для переносимости на RISC-процессоры, где только два кольца защиты, т.к. однобитовый флаг режима).
- Для того, чтобы прикладной процесс все-таки мог вызвать системную процедуру, используются шлюзы – специальные дескрипторы, содержащие вместо адреса и размера сегмента ссылку на селектор и смещение в некотором другом сегменте.

Защита страниц памяти в Intel

В элементе таблицы страницы есть два бита:

- Флаг права доступа: **W** или **R** – разрешена запись или только чтение
- Флаг привилегий текущего режима: **S** (supervisor) для 0, 1 и 2 уровня или **U** (user) для 3 уровня – устанавливается ОС
  - только в режиме **S** разрешено выполнение привилегированных команд – таких, как загрузка таблицы дескрипторов или слова состояния программы. Это еще один вид защиты – защита команд процессора
- Таким образом, защита страниц частично дублирует возможности защиты сегментов – для большей надежности и для тех ОС, которые не используют сегментную организацию памяти

Использование возможностей аппаратной защиты Intel в Windows

- Ядро и драйверы устройств выполняются в 0 кольце защиты, остальной код ОС и приложения – в 3 кольце (для переносимости на RISC-процессоры, у которых только два кольца защиты – однобитовый флаг режима).
- Каждый процесс получает адресное пространство размером 4 GB – шесть совпадающих друг с другом сегментов: код, данные и стек для 3 кольца защиты и то же для 0 кольца
- Младшая половина сегмента – это пространство собственных адресов процесса. Старшая – системное пространство адресов, которое содержит одни и те же страницы для всех процессов; обратиться к ним можно только из 0 кольца
  - последнее ускоряет обработку аппаратных прерываний в 3 кольце: обработчик работает в 0 кольце, но страница, его содержащая, при возникновении прерывания находится в текущем адресном пространстве, и не требуется перезагрузка каталога страниц – нужно только перезагрузить селекторы в сегментных регистрах кода и стека
    - Это делается аппаратно в ходе генерации прерывания
    - Так существенно уменьшается время переключения контекста

Функции подсистемы защиты в ОС

1. Разграничение доступа – см. слайды 12 -13
2. Обеспечение безопасности: *идентификация* (опознавание) и *аутентификация* (подтверждение истинности субъекта) при *авторизации* пользователей (допуска их к выч. системе с определенными правами)
3. Аудит - регистрация потенциально опасных событий в *журнале безопасности*
4. Управление политикой безопасности – ее задает администратор с помощью средств определения объектов защиты и раздачи прав пользователям
5. Криптография – шифрование защищаемой информации. Эти методы рассматриваются в разделе вычислительной науки «Защита информации»
6. Сетевые функции – в сетевых ОС: защита обмениваемой информации и защита от сетевых атак и вирусов

Обеспечение безопасности при авторизации пользователей

Имя – замок (идентификация), пароль – ключ (аутентификация). Пароли хранятся в ОС в зашифрованном виде, причем пароли пользователей не должен знать администратор

Меры повышения стойкости символьных паролей против их взлома:

- регулярная смена паролей (раз в месяц, в неделю; в пределе – после каждого сеанса – так называемые одноразовые пароли)
- длина 10 –14 символов различных регистров
- блокировка терминала или учетной записи (account) пользователя после нескольких неудачных попыток ввода пароля
- генерация случайного пароля ОС

Аппаратные средства аутентификации повышают степень защиты:

- Ключ на внешнем носителе: он может быть гораздо длиннее запоминаемого пароля; это используется в комбинации с обычным паролем. Наиболее стойкий ключ – на интеллектуальной пластиковой карте (Smart Card). Она содержит микропроцессор, который проверяет правильность пароля и стирает ключ после превышения допустимого максимума неправильных попыток ввода пароля.
- Устройства проверки биометрических характеристик пользователя: голоса, отпечатков пальцев, почерка и т.д. Этот метод дорог и имеет ненулевую вероятность ошибки.

(Вопрос 5)

Аудит подсистемы защиты в ОС

- Аудит - регистрация потенциально опасных событий в *журнале безопасности*
- Пользователи – *аудиторы,* обладающие правом чтения этого журнала, могут анализировать состояние безопасности, отличать случайные нарушения от атак
- В идеале, администратор системы не должен иметь прав аудитора, но это обычно невозможно обеспечить в большинстве ОС (включая Windows и Unix)

(Вопрос 6)

Подсистема защиты в Windows (1)

- Много объектов защиты: защищается все, включая семафоры
- Много типов субъектов
  - 6 предопределенных типов
  - Временные группы
- Много (до 22) видов операций доступа
  - 6 стандартных
    - Удаление объекта
    - Получение атрибутов защиты объекта
    - Изменение атрибутов защиты объекта
    - Изменение владельца объекта – на себя
    - Получение и изменение параметров аудита в отношении объекта
    - Синхронизация – ожидание изменения состояния объекта
  - Специфичные
    - Напр., для файла: Чтение. Запись. Добавление информации в конец. Выполнение. Получение атрибутов. Изменение атрибутов.

Подсистема защиты в Windows (2)

- Права доступа к объектам
  - Каждому методу доступа соответствует право на его применение
  - Кроме того, поддерживаются *обобщенные (generic)* или *отображаемые (mapped)* права – четыре вида наборов стандартных и специфичных прав: GENERIC_READ, GENERIC_WRITE, GENERIC_EXECUTE, GENERIC_ALL.
- Маркер доступа - объект, содержащий всю необходимую информацию для принятия решений о разрешении доступа субъекту (процессу или потоку):
  - идентификатор пользователя
  - идентификаторы групп, в которые входит пользователь
  - привилегии пользователя
  - идентификатор сеанса работы
  - атрибуты защиты по умолчанию для создаваемых пользователем объектов
  - служебную информацию

Подсистема защиты в Windows (3)

- Дескриптор защиты - служебная структура данных объекта, содержащая атрибуты его защиты:
  - идентификатор владельца объекта
  - идентификатор первичной группы владельца
  - список избирательного контроля доступа (DACL – discretionary access control list) – столбец матрицы защиты, соответствующий объекту
  - системный список контроля доступа (SACL –system access control list) – используется при генерации сообщений аудита
- Принятие решения о доступе – в результате сравнения маркера доступа субъекта с дескриптором защиты объекта
- Общая характеристика подсистемы защиты в Windows vs. Unix
  + Гибкость: много возможностей
  – Избыточность: слишком много возможностей
  – Противоречия и ошибки
  – Плохая документированность

Вопросы к лекции

1. Почему бы не строить модель: $S \times Y \rightarrow A$, т.е. специфицировать все тройки: *субъект-объект-право_доступа* ?
2. Сравним два этих подхода: например, в W2k можно назначать большее число классов полномочий для доступа к файлам, чем в Unix, и, в частности, разрешать произвольному пользователю доступ к отдельному файлу. К каким противоречиям это может привести?
3. Опишите в терминах матрицы доступа способ защиты памяти при ее распределении непрерывными разделами (например, в OS/360).
4. Почему не разрешается назначать владельцем некоего третьего субъекта ?
5. Какой метод повышения стойкости аутентификации пользователей представляется вам наиболее перспективным в ближайшем будущем?
6. Какой субъект ОС может иметь право записи в журнал аудита? Право удаления (очистки) журнала? Что должно происходить в случае переполнения журнала?

Резюме по курсу

Что вы узнали:

- Базовые принципы построения и работы ОС – они не меняются последние 40 лет (по-видимому, вечные)
- Их воплощения в двух популярных ОС
  - Windows
  - Unix/Linux – основы всех без исключения новых ОС (по-видимому, и будущих тоже)
    - К сожалению, MAC OS не рассматривалась ☹
  - Знание этих ОС позволит эффективно их использовать
- ОС – пример большой и сложной программной системы
  - Ее организации можно подражать в других больших программах
- Акцент на «горячих точках» программной инженерии, связанных с ОС
  - Параллельное программирование
    - для многопроцессорных и многоядерных систем
    - для распределенных систем
  - Обеспечение отказоустойчивости

Организация ОС как большой программной системы

Два варианта модульного построения:

- Монолитная ОС: высокая производительность и реактивность
- Микроядро: хорошая гибкость, масштабируемость

Реальные системы – компромисс:

- Windows: в основном микроядро
- Unix/Linux: в основном монолитная

NB: компромисс – tradeoff – один из основных инженерных принципов

Типичные структуры данных ОС

- Дескрипторы ресурсов ОС
- Одновходовые таблицы – реализация бинарных отображений виртуальных ресурсов на физические в виде записей или одномерных массивов
  - напр., таблицы страниц
- Простые списки
  - напр., список открытых файлов
- Упорядоченные списки – очереди запросов, ресурсов, прерываний, сообщений; буферы, кэши, … - очереди разных видов:
  - FIFO
  - LIFO (стеки)
  - LRU и его приближения

Типичные алгоритмы ОС

- Синхронизация и борьба с тупиками
- Отложенные действия
  - для обеспечения атомарности, напр., транзакций
  - для сглаживания «рывков», напр., очередь отложенного вытеснения страниц
  - вездесущие буферизация и кэширование
- Упреждающие действия

Ваша оценка курса?

- Что повторяло известное из прежних курсов?
- Что было новым?
- Что представляется более полезным, что менее?
- Что изложено недостаточно понятно?