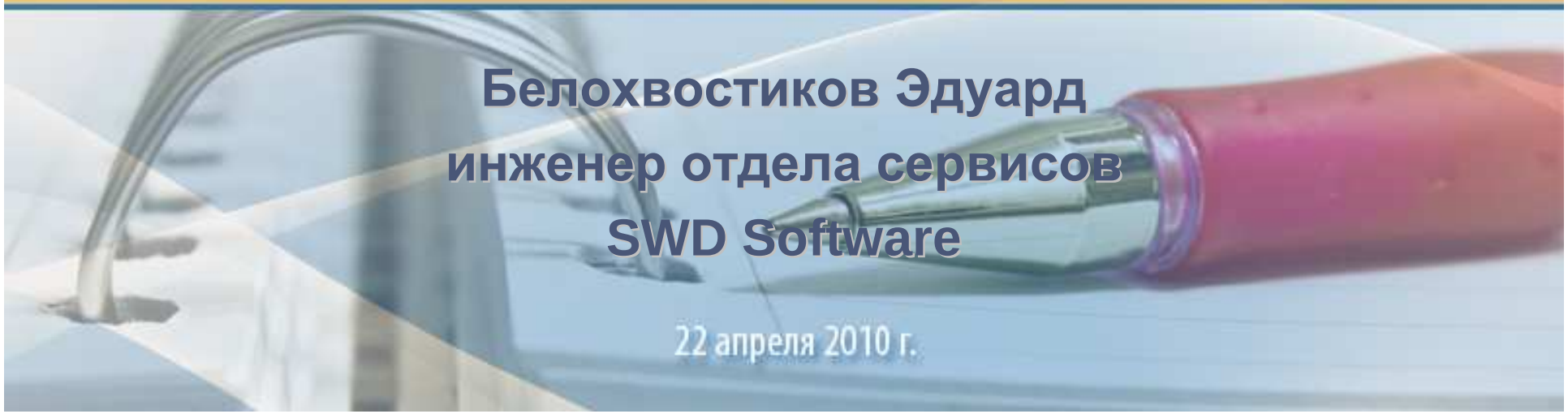
XI Международная конференция 2010
QNX Россия

Технологии будущего для реального времени

Построение человеко-машинных интерфейсов на базе технологии QNX Aviage HMI



Белохвостиков Эдуард
инженер отдела сервисов
SWD Software

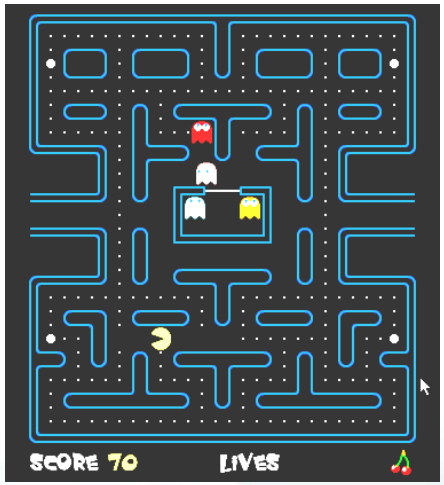
22 апреля 2010 г.

Пользовательский интерфейс

- Высокая интерактивность
- Масштабируемость
- Универсальность
- Надежность
- Портируемость
- Высокая скорость разработки
- Эффективное использование памяти и процессора

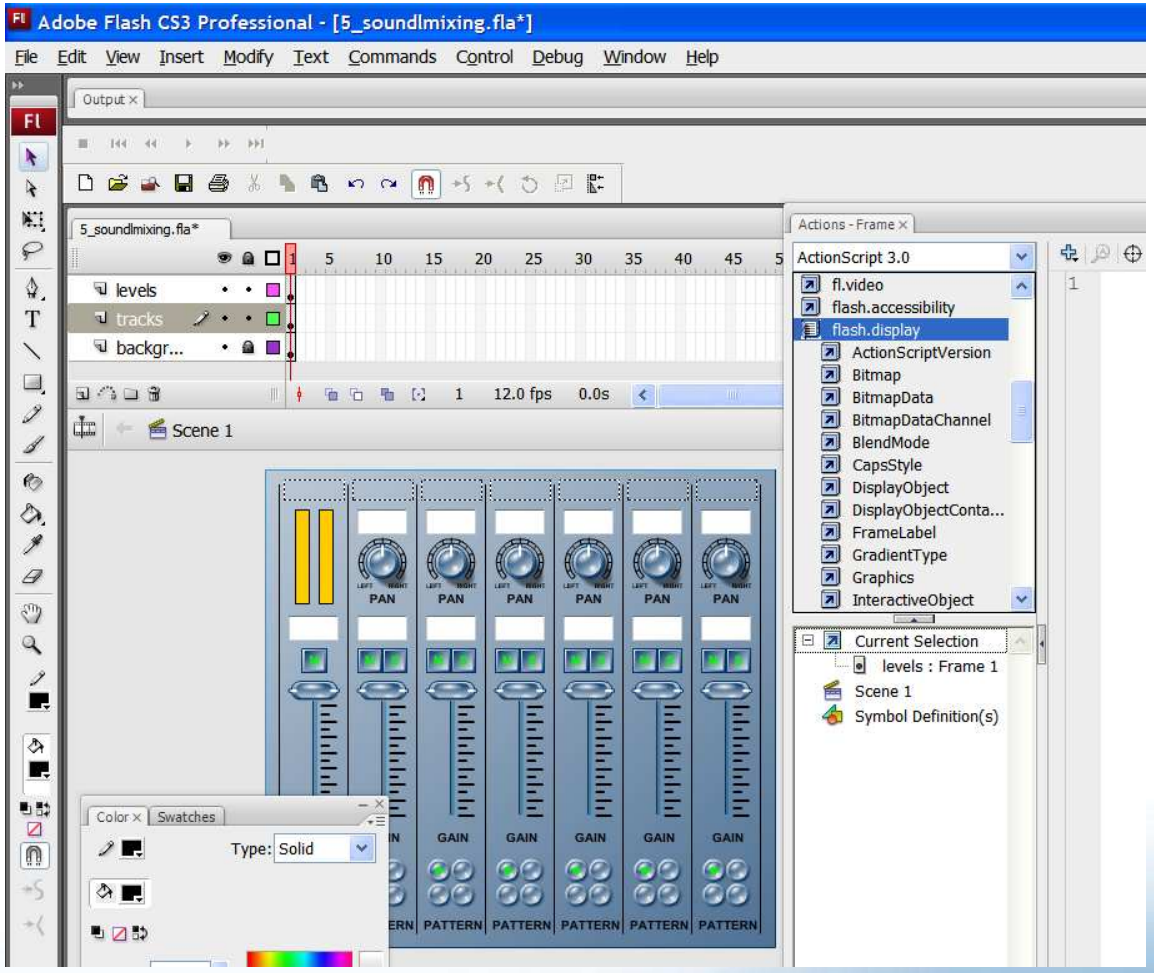
Почему технология Flash?

- Flash-технология де факто является стандартом HMI-интерфейса
  - Flash-плеер установлен 99.3% *всех* пользователей Интернета*
  - Во всем мире насчитывается более миллиона графических дизайнеров
- Во Flash-технологии реализуется обещание языка Java: "Написанное однажды - работает везде"
  - Нет проблем зависимости от библиотеки классов
  - Контент, разработанный для среды веб или PC, без изменений работает во встраиваемых системах
  - Встраиваемый плеер FlashLite использует меньше памяти и обеспечивает более быструю визуализацию
- Flash-технология идеальна для создания HMI-интерфейсов
  - Графика – основа технологии
  - Используется промышленными дизайнерами для создания диалоговых интерфейсов



* Приведенные значения являются оценкой компании Adobe

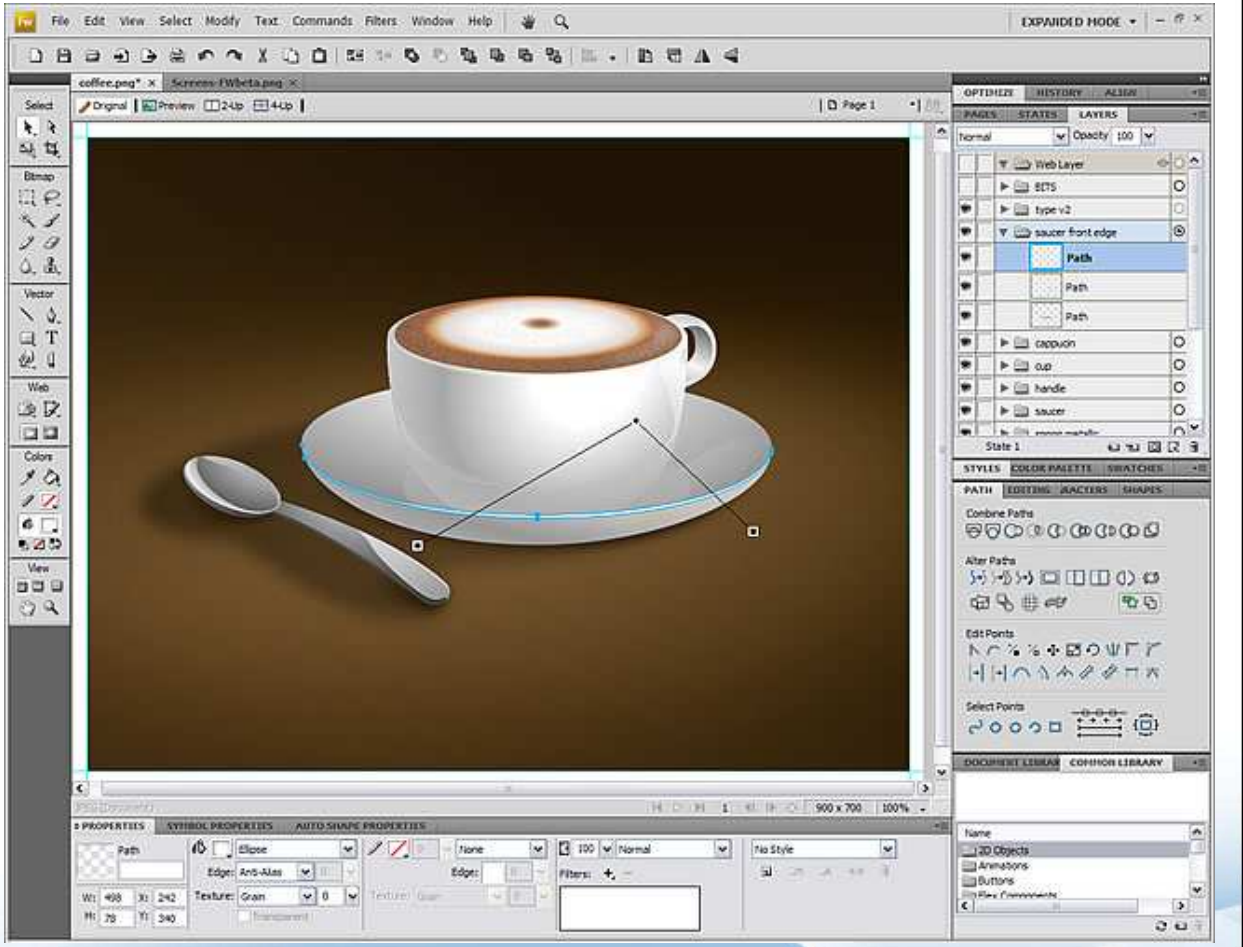
Основные характеристики технологии Adobe Flash



- В разработке используется “шкала времени”
  - Скорость анимации можно задавать числом кадров в секунду
  - Шкала времени управляет процессом анимации
- Для реализации действий в сценарии используется язык ActionScript
  - Похож на язык JavaScript
  - К объектам применяется процесс "сборки мусора"
  - Исполнение реализовано одним потоком
- Кодирование производится в среде RAD
  - Действия ассоциируются с событиями
  - Код может быть "разбросан" по всей модели

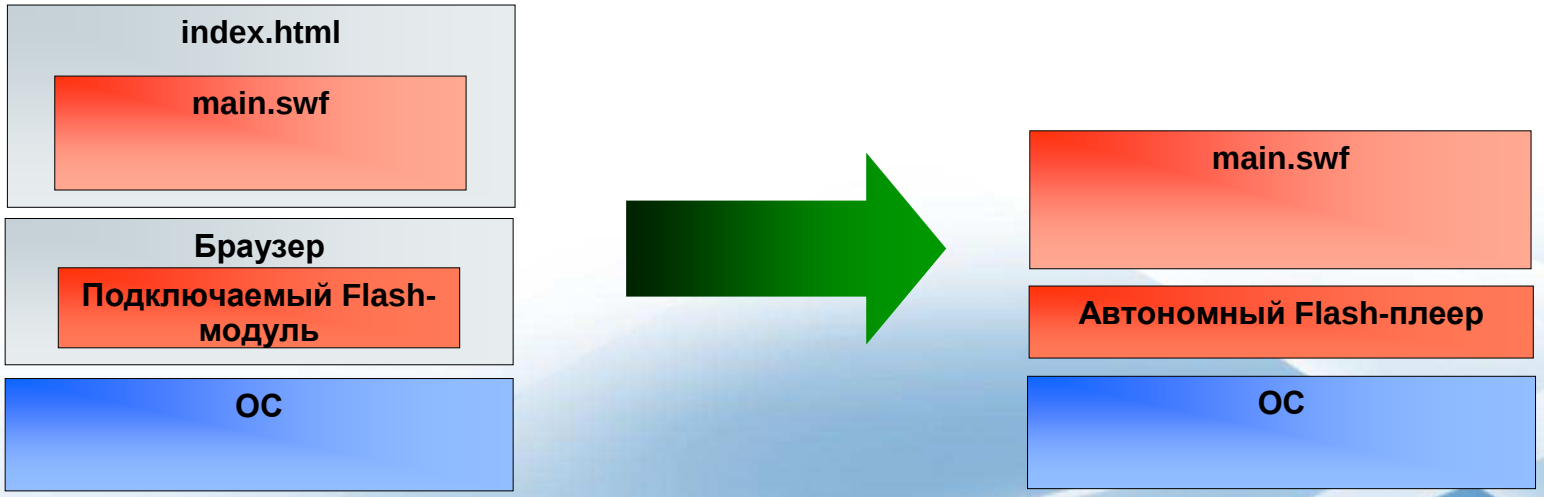
Среда разработки Adobe

- Управление перемещением объектов, состоянием и взаимодействием в "фильме" происходит под управлением шкалы времени
- Имеется множество предварительно созданных сценариев (эффекты, переходы, построение промежуточных отображений – tweens и т.д.)
- Поведение определяется как сценарием (кодом), так и шкалой времени
- Все графические ресурсы изначально относятся к типу векторной графики



Flash в качестве HMI-администратора

- Flash обычно "живет" в среде браузера
- Во встраиваемых приложениях нужно использовать Flash-контент в качестве основной графической среды
  - Управление панелью отображения
  - Основной уровень пользовательского интерфейса
  - Запуск и управление любыми резидентными приложениями



Flash и сторонние приложения

- Расширение fscommand2()
- Расширение get_url()

- Разработка собственного класса на C/C++, доступного из ActionScript.

Поддержка мультимедиа

Поддержка вывода звука

- Технологии PCM и ADPCM, встраиваемый MP3.

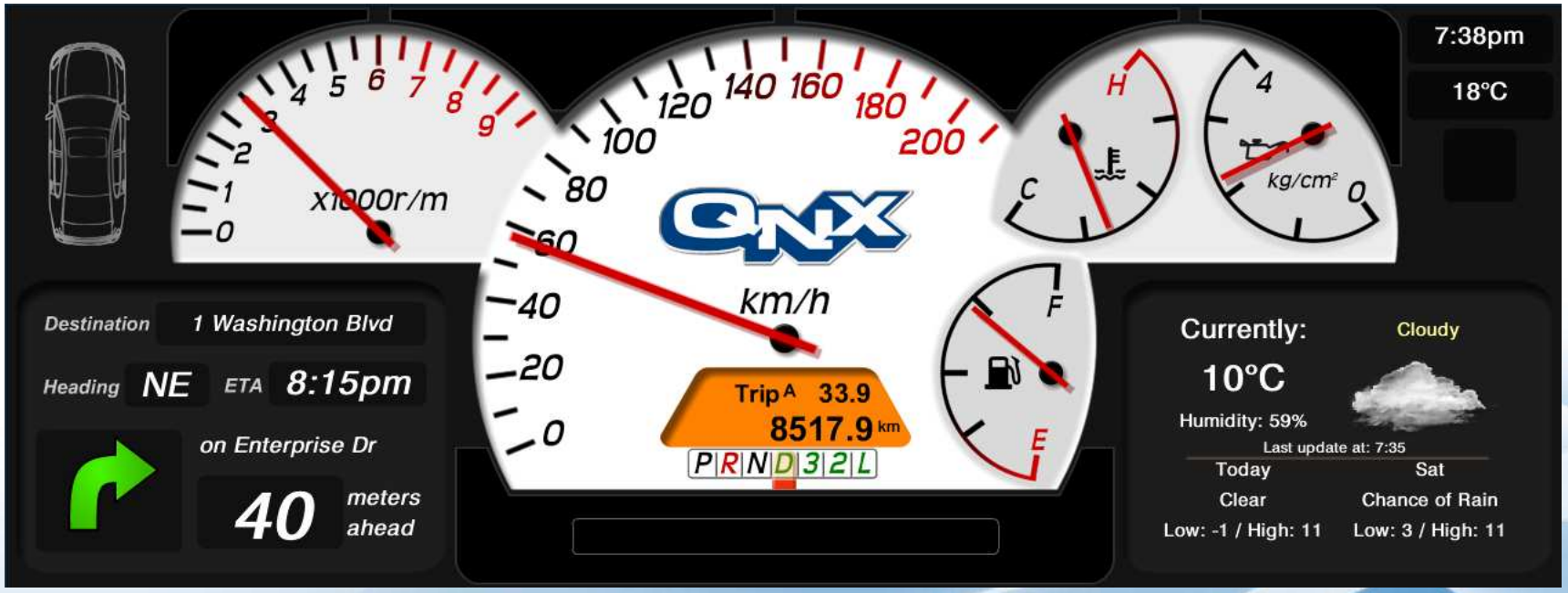
Поддержка вывода изображений и видео

- Форматы изображений: PNG, JPEG, GIF, BMP, SGI, TGA, анимированный GIF.
- Flash-видео (FLV) с использованием кодеков On2 и Sorenson.
- Сглаживание и распаковка видеоблоков.

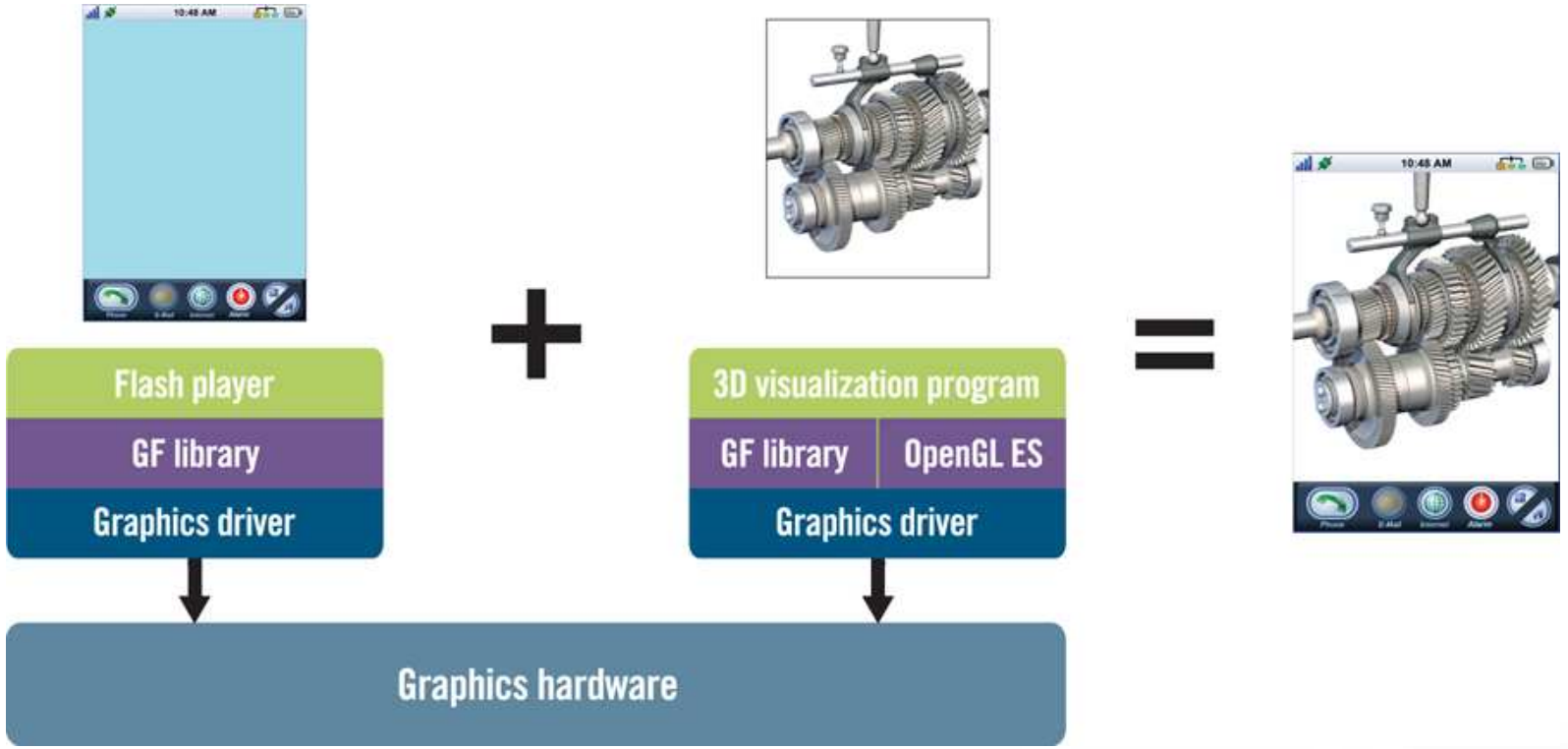
Программный интерфейс к медиаконтролеру (Aviage Multimedia Suite)

Flash-код и резидентный код

- Разделение графических элементов обеспечивает изолированность процессов
- С разделенными процессами легче гарантировать высокую готовность системы
- Изображение на дисплее составляется из слоев
  - Аппаратные слои (обеспечиваются графическим контроллером)
  - Программные слои (обеспечивается платформой OpenKode или аналогичной)



Слияние Flash- и резидентных приложений

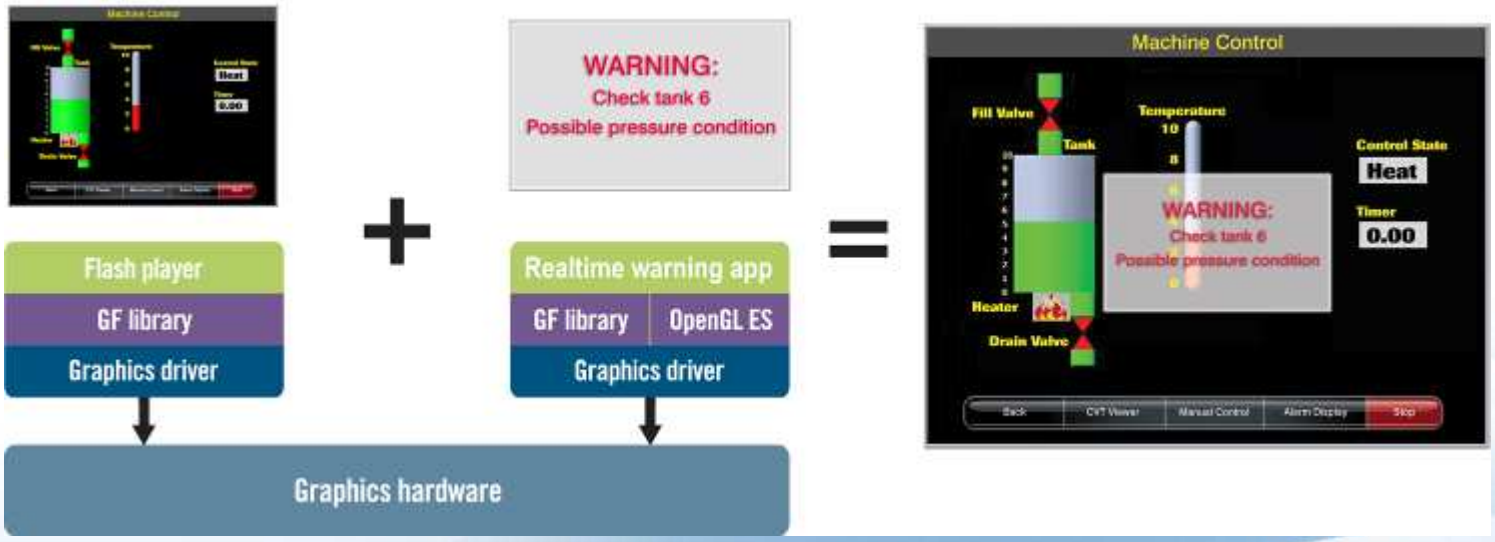


- Причины для слияния резидентного и Flash-кода
  - Уже есть существующий графический код
  - Есть код с 3D-функциями (на базе OpenGL ES)
  - Необходима производительность выше, чем может предоставить Flash-код

Технология слоев

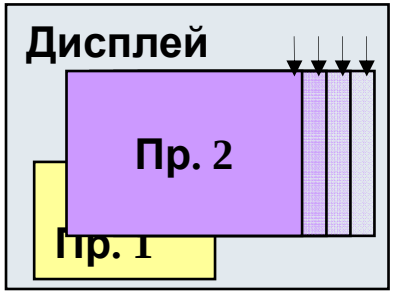
Способы объединения контента:

- изменение позиции
- поворот
- масштабирование
- альфа-смешивания
- хроматческий ключ



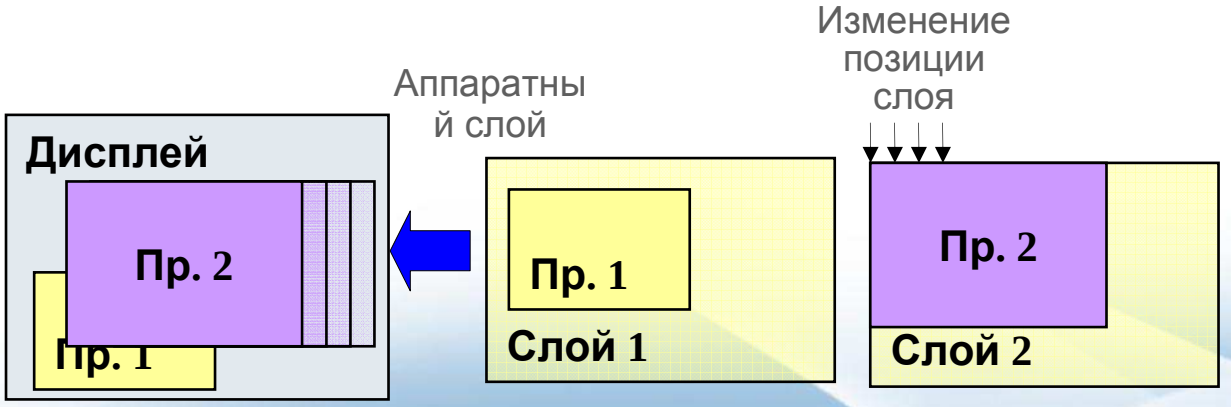
Перемещение объекта

Традиционный подход



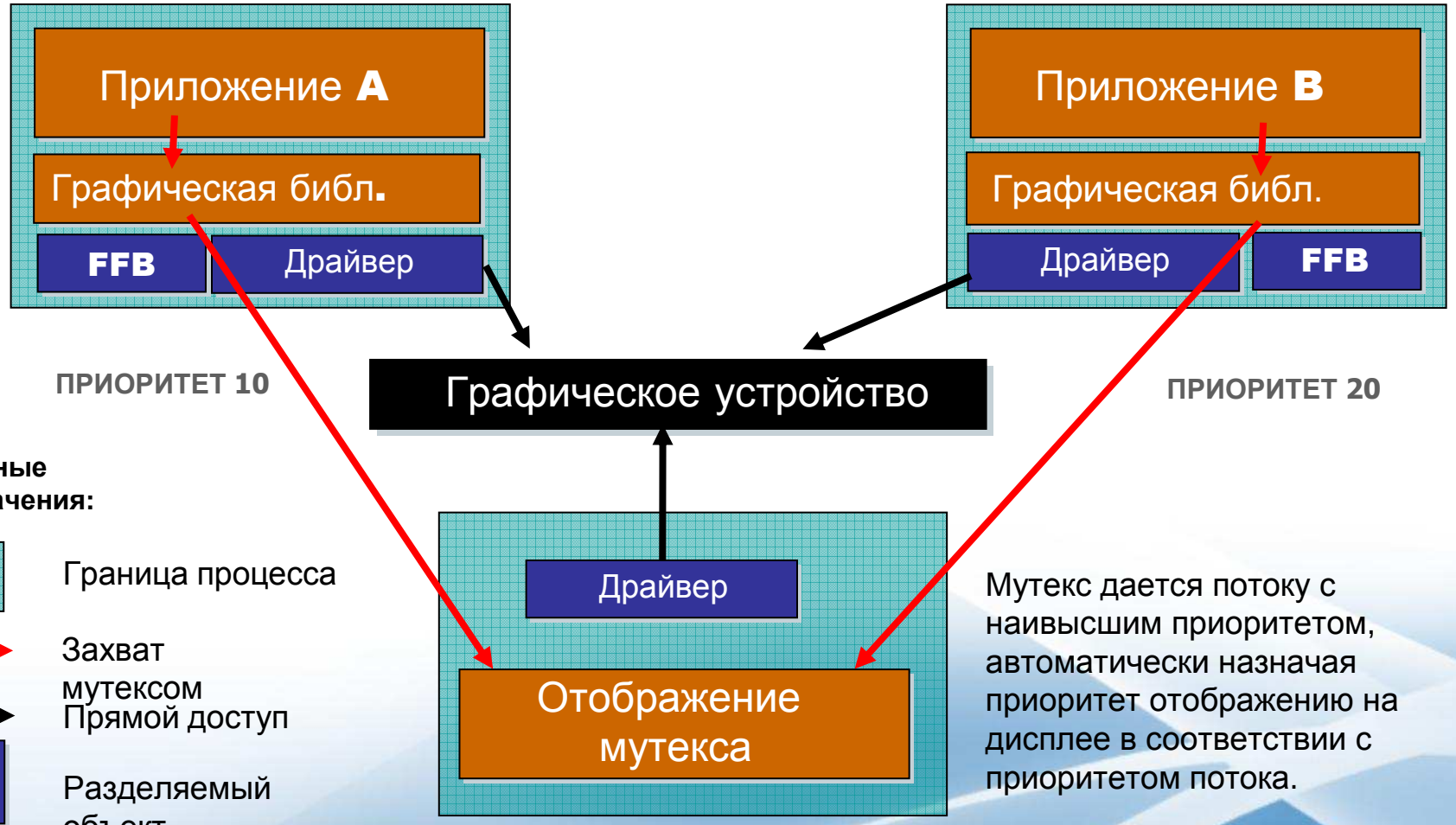
Перерисовка дисплея после каждого движения

Индивидуальный слой

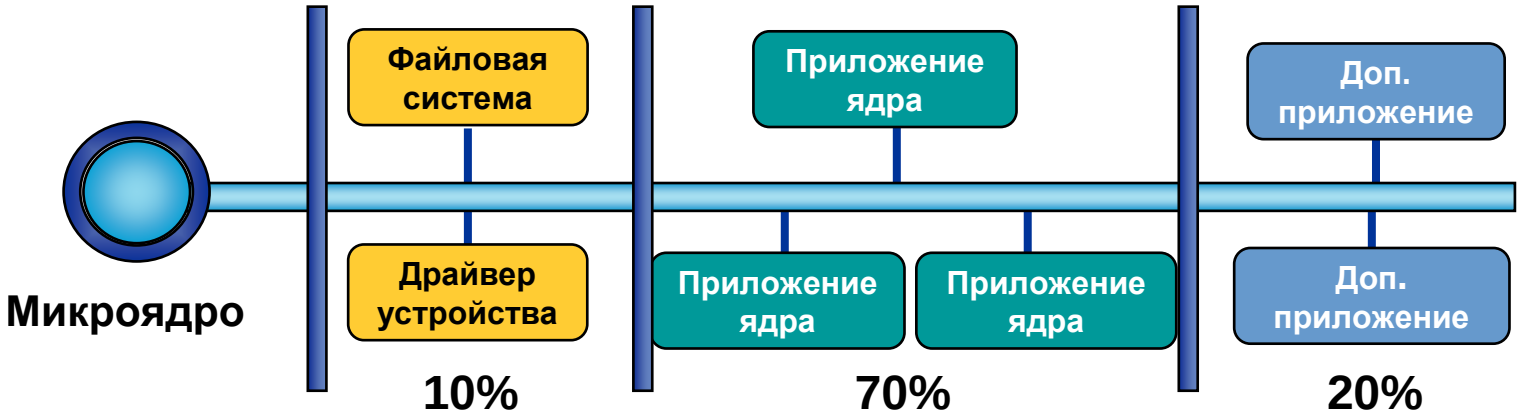


Предсказуемое время реакции системы

Координация работы нескольких графических приложений

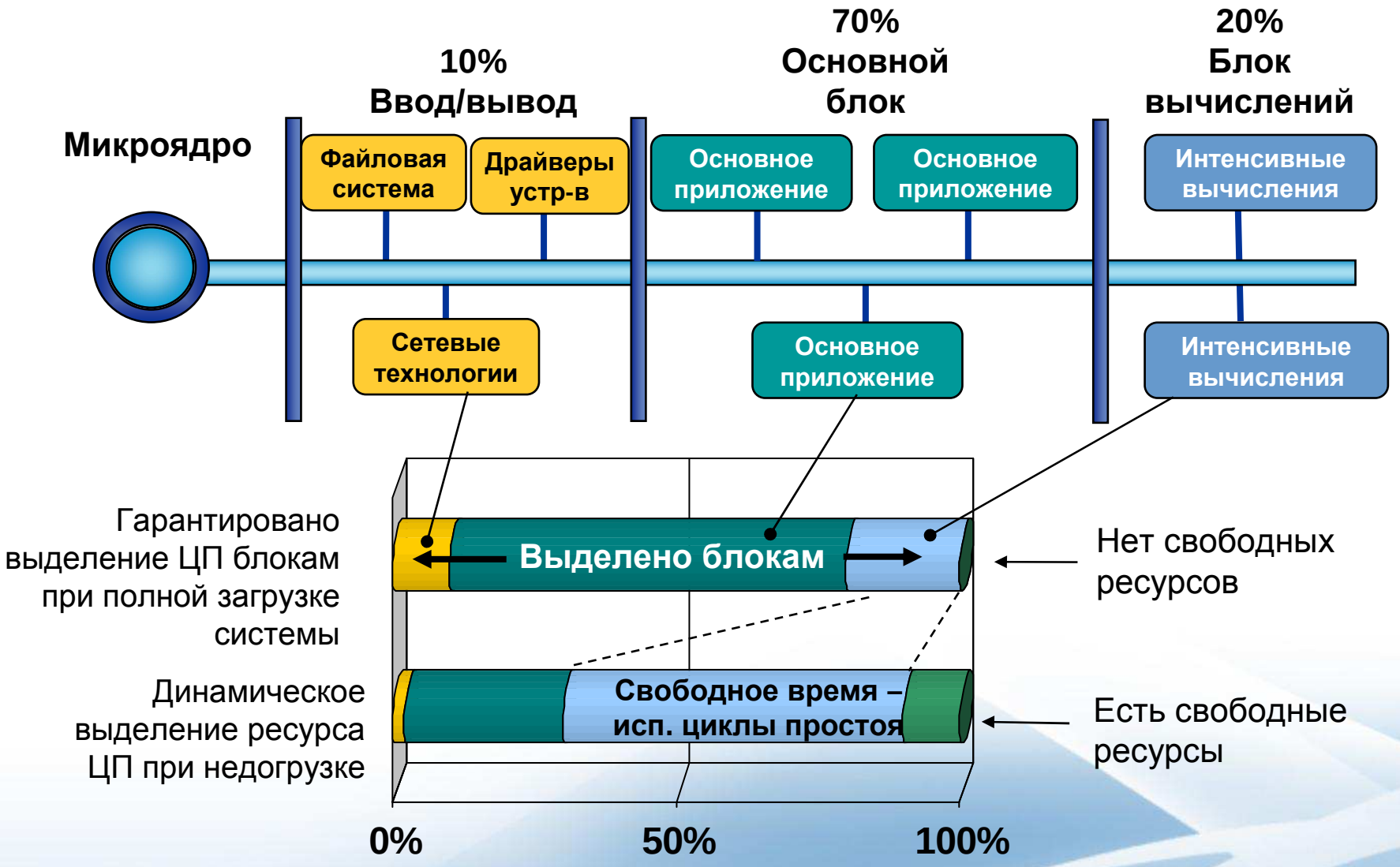


Декомпозиция по времени



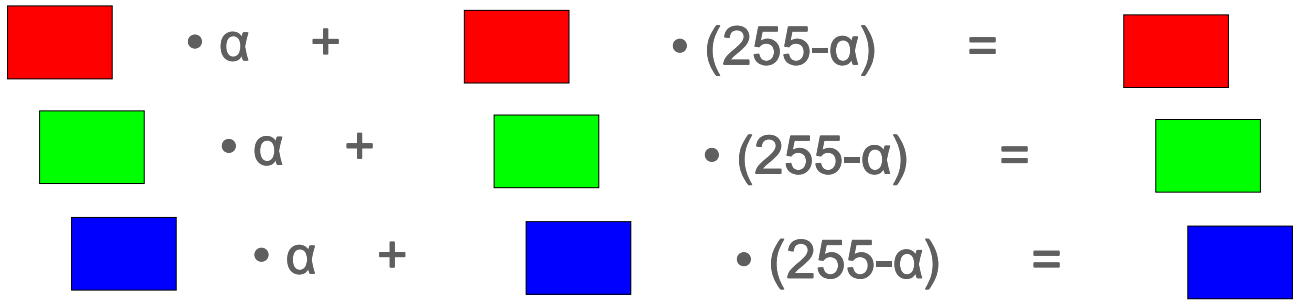
- ОСРВ предоставляет базовую структуру
  - Инкапсуляция приложений и служб ОС с передачей сообщений
  - Аппаратная защита памяти для обеспечения надежности и безопасности.
- При декомпозиции по времени предоставляются безопасные блоки и гарантированное время доступа к ЦП
  - Блок состоит из группы процессов и потоков
  - Блоку назначается процент времени ЦП, усредненный по временному окну
  - Осуществляется перекрытие существующего процесса планирования потоков

Адаптивное квотирование QNX

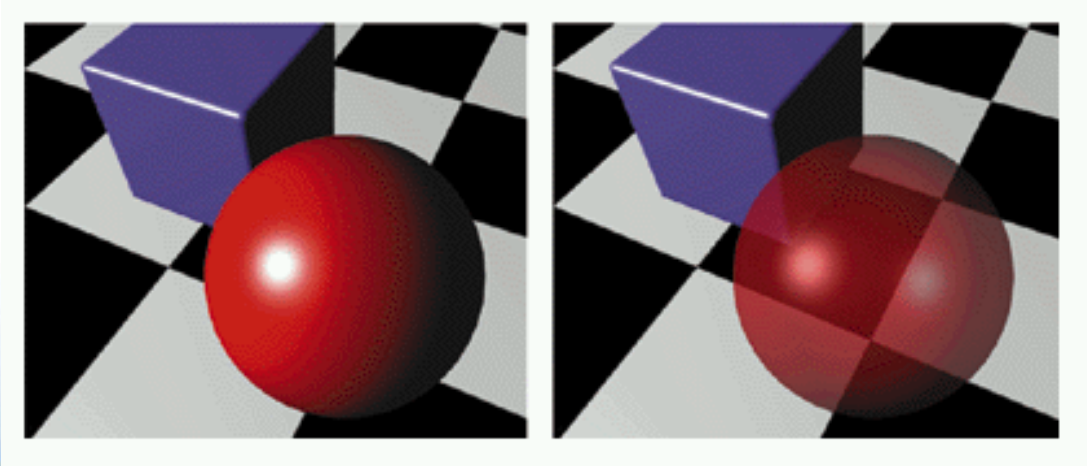


Специфические рекомендации для Adobe Flash

- Минимально использовать alpha-смешивание (прозрачность)

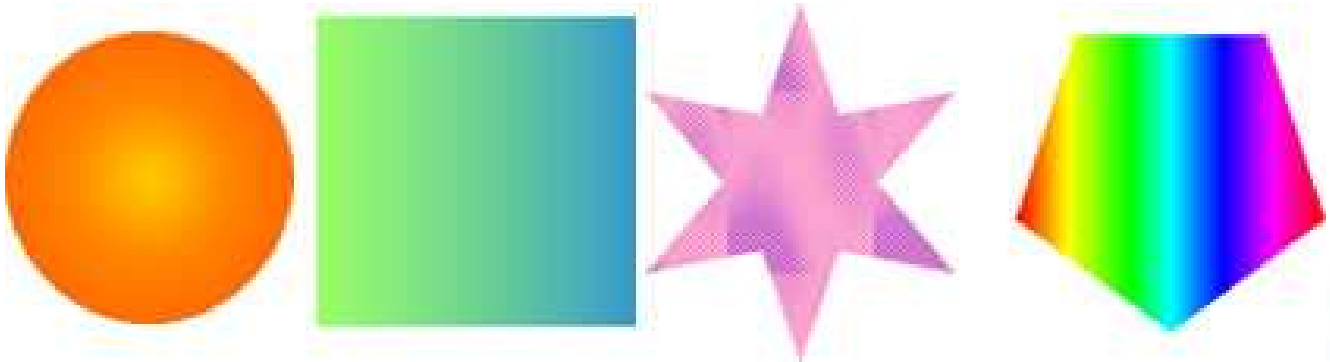


- Байтовые операции на точку: 6 загрузок, 6 умножений, 3 сложения, 3 сохранения
- Отдельный alpha-канал (дополнительно): 1 загрузка, 1 вычитание



Оптимизация Adobe Flash

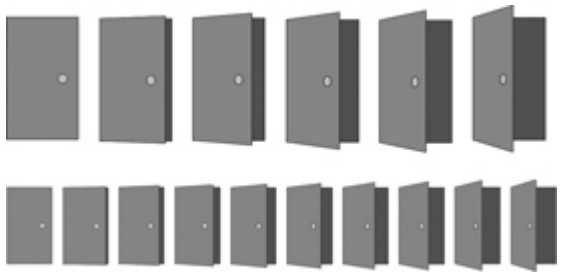
- Создание градиента требует интерполяции между несколькими цветами
  - Определение цветов на каждой стороне точки
  - Вычисление интерполяционного процента
  - Перемножение процентов по каждой паре цветов
- Adobe предоставляет усложненные градиенты
  - Радиальный, линейный (несколько точек), битовый образ, радуга, угловой и т.д.



- Нужно выбирать простейший, приемлемо выглядящий градиент
- Использовать кеширование битового образа, если это возможно

Оптимизация Adobe Flash

- Использовать минимально допустимую частоту смены кадров
  - 40-60 дает плавную анимацию
  - 20-25 выглядит приемлемо
  - 15 является типичной рекомендованной «нижней границей» для веб
  - 10-12 может быть наилучшим выбором для встраиваемых систем



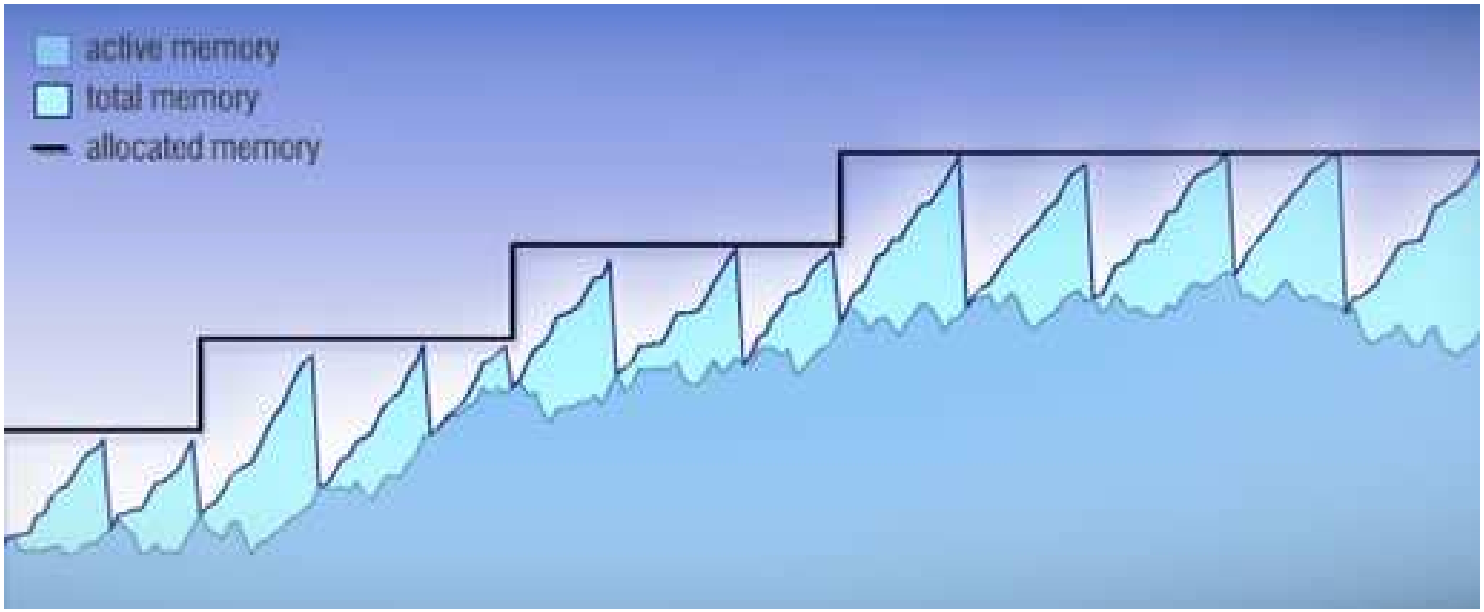
- Частота кадров определяет скорость проигрывания «фильма» и число генерируемых переходов в сек.
- Больше кадров = больше ресурсов ЦП и больше памяти
- Использование нескольких SWF-фильмов, используемая частота кадров определяется вызовами
- Может привести к непредвиденной частоте смена кадров
- Простейшее решение: использовать в проекте стабильную частоту смены кадров

Оптимизация Adobe Flash

- Способы оптимизации
  - Проверить, что глубина цвета согласуется с графикой (555/565/8888)
  - Оптимизировать битовые образы для отображения
  - Использовать расширения для реализации визуализации и прокрутки вне процесса вывода изображения
  - Сложную обработку реализовывать не на языке ActionScript, а в виде резидентного кода; это приведет к более быстрому исполнению задачи

- Задержки в работе ActionScript будут уменьшать скорость обновления HMI-интерфейса (исполнение одним потоком)
  - Adobe Player имеет ограничения на случай «выхода сценария из-под контроля»; если сценарий исполняется более нескольких секунд, не возвращая управления, его работа прекращается
  - Длительные по времени операции можно переместить во внешний многопоточный резидентный код, вынеся его за границы ActionScript

Использование ресурсов памяти



- ActionScript является языком с автоматическим сборщиком мусора
  - Система «сборки мусора» использует счетчик ссылок и метки, а также систему «зачистки»
  - Система «сборки мусора» запускается периодически на основе эвристического анализа памяти
  - Очистка памяти отложена, она не выполняются немедленно

Управление памятью HMI-интерфейса

- Выделение фиксированного размера динамически выделяемой памяти в приложениях Adobe
  - Определить параметр dynamic_mempoolsize=<*SIZE_IN_BYTES*>
  - Adobe рекомендует использовать начальное отношение “контент – динамическая память» равным 1:15
- Проверка использования памяти из приложения:
  - Общий объем памяти, доступной плееру
  - Объем свободной памяти, доступной плееру
- Использование, если это возможно, декомпозиции памяти для резервирования или гарантированного выделения памяти для других критических процессов в системе

Использование ресурсов памяти

- Помогайте работать системе «сборки мусора» Flash-приложения
  - Устанавливайте ссылку на объект в нулевое значение, когда объект перестает использоваться
  - Явным образом удаляйте «слушающий» объект перед его выгрузкой
  - Ограничивайте использование глобальных переменных

- Техника экономии памяти
  - При кешировании битового образа воспроизведение происходит быстрее, но требуется больше памяти
  - Внимательнее относитесь к шрифтам, используйте системные, векторные и разделяемые шрифты.

fonts

Использование резидентного кода

- Нужна производительность резидентных графических приложений
- Нужны 3D-функции
- Нужен вызов существующего резидентного кода
- Нужна изоляция процесса
- Нужна многопоточность (и/или нужно предотвращать блокировку HMI-интерфейса)
- Нужны функциональные возможности низкого уровня, не доступные через ActionScript
- Нужно выполнять интенсивные в вычислительном отношении или затратные по времени задачи
- Нужно иметь предсказуемое поведение

Шлюз от Flash- кода к резидентному коду

- Реализация расширений функции fscommand2()
  - *fscommand_direct*() – C-процедура для связи с ActionScript *fscommand*(“mycommand”, “arg1”, “arg2”, …);
- Реализация расширений функции geturl()
  - *geturl*() – добавление собственных личных типов url
  - Работает только для элементов, естественным образом выраженных как URL-ссылки внутри приложения (например, проиграть медиаресурс “mp3://audioprompt.mp3”)
- Разработка собственного класса на языках C или C++
  - на языке ActionScript определяется интерфейс
  - Реализация обеспечивается резидентным кодом (на языке C/C++)

QNX Aviage HMI

- Высокая скорость разработки
- Динамичный пользовательский интерфейс
- Объединение Flash-контента и резидентного кода
- Ориентированность на встраиваемые системы
- Создание интерфейса к Flash-приложению
- Возможность замены базовых компонентов без изменения пользовательского интерфейса и наоборот

Спасибо!

Белохвостиков Эдуард
eduard.b@swd.ru