Введение в конструирование программного обеспечения

Франко Гасперони
gasperon@act-europe.fr
http://libre.act-europe.fr/Software_Matters
перевод: Владислав Козловский
dbdeveloper@rambler.ru

Домашняя страница курса

- **http://www.ada-ru.org/slides**
  - Здесь находятся все слайды курса (PDF и PowerPoint)

Местонахождение оригинального курса

- **http://libre.act-europe.fr/Software_Matters**
  - Здесь находятся все слайды оригинального курса (PDF и PowerPoint)

Уведомление об авторском праве

- **© ACT Europe согласно GNU Free Documentation License**
- **© Владислав Козловский (перевод) согласно GNU Free Documentation License**
- Позволяется копировать, распространять и/или модифицировать этот документ согласно условиям GNU Free Documentation License, Версии 1.1 или более поздней, опубликованной Free Software Foundation, при условии упоминания автора оригинала и переводчика, а также сохранении ссылки на первоисточник (http://libre.act-europe.fr/). Полный текст лицензии доступен по адресу:
- **http://www.fsf.org/licenses/fdl.html**

Цель данного курса

- **Помочь создавать программные системы, которые более:**
  - Надёжны
  - Гибки
  - Просты в разработке

- **Показать узкие места и проблемы C-подобных языков программирования**
  - C, C++, Java

- **Сравнить способы структурирования программ**
  - Функционально-ориентированный
  - Объектно-ориентированный
  - Проблемы, возникающие в обеих подходах

- **Показать, как Ada 95 удачно отвечает на эти вопросы**
  - Как применять в других языках принципы конструирования, заложенные в Аду

Интересные книги

- **Адское программирование**
  - Александр Гавва (электронная версия на http://www.ada-ru.org)
- **Programming in Ada 95**
  - by John Barnes (Addison Wesley)
- **High Integrity Ada: The SPARK Approach**
  - by John Barnes (Addison Wesley)
- **Object-Oriented Software Construction**
  - by Bertrand Meyer (Prentice Hall)
- **Objects Unencapsulated: Java, Eiffel, and C++**
  - by Ian Joyner (Prentice Hall)
- **C Traps and Pitfalls**
  - by Andrew Koenig (Addison Wesley)
- **Effective C++**
  - by Scott Myers (Addison Wesley)

Интересные ссылки

- **http://www.ada-ru.org**
  - Ада по-русски. Сайт русскоязычного сообщества языка Ада.
- **http://www.fsf.org**
  - Сайт Фонда Свободного ПО (the Free Software Foundation) и проекта GNU
- **http://libre.act-europe.fr**
  - Interesting Free Software projects written in Ada 95
- **http://adapower.com**
  - Очень интересный сайт посвященный языку Ада, с огромным колличеством информации и учебных пособий
- **http://www.adaic.com/whyada/ada-vs-c/cada_art.html**
  - Сравнение цены разработки с использованием языков C и Ada

Предполагается, что:

- **Вы интересуетесь разработкой программного обеспечения**
- **У вас в активе есть программы, написанные хотя бы на одном из императивных языков**
  - Таких как Ada, C, C++, Eiffel, Fortran, Java, Pascal, …
- **Вы знакомы (хотя бы поверхностно) с языком C**
  - … для секции раскрывающей проблемы и подвохи C-подобных языков программирования

Основы программостроения

Ваш опыт разработки программного обеспечения (ПО)

- **Вспомните самый крупный проект по разработке ПО, в котором Вам довелось участвовать**
- **Как его разрабатывали?**
  - Какой использовали процесс создания ПО?
  - Какой язык программирования?
  - Какие инструменты?
  - Использовалась ли система контроля версий?
- **Долго ли планируется эксплуатировать созданное программное обеспечение?**
  - Кто будет сопровождать (исправлять, изменять или адаптировать) его на протяжении всего этого времени?

Малые программные системы...

- **Понятны одному человеку**
- **Могут быть полностью переписаны с нуля для**
  - Исправления ошибок или добавления новых возможностей
  - Переноса на другую платформу
- **Любые средства хороши для их создания**
- ***Стиль написания зависит от программиста*** ☺
- **Обычно любая программа объемом менее 10 000 строк кода считается малой**

Средние/Большие программные системы…

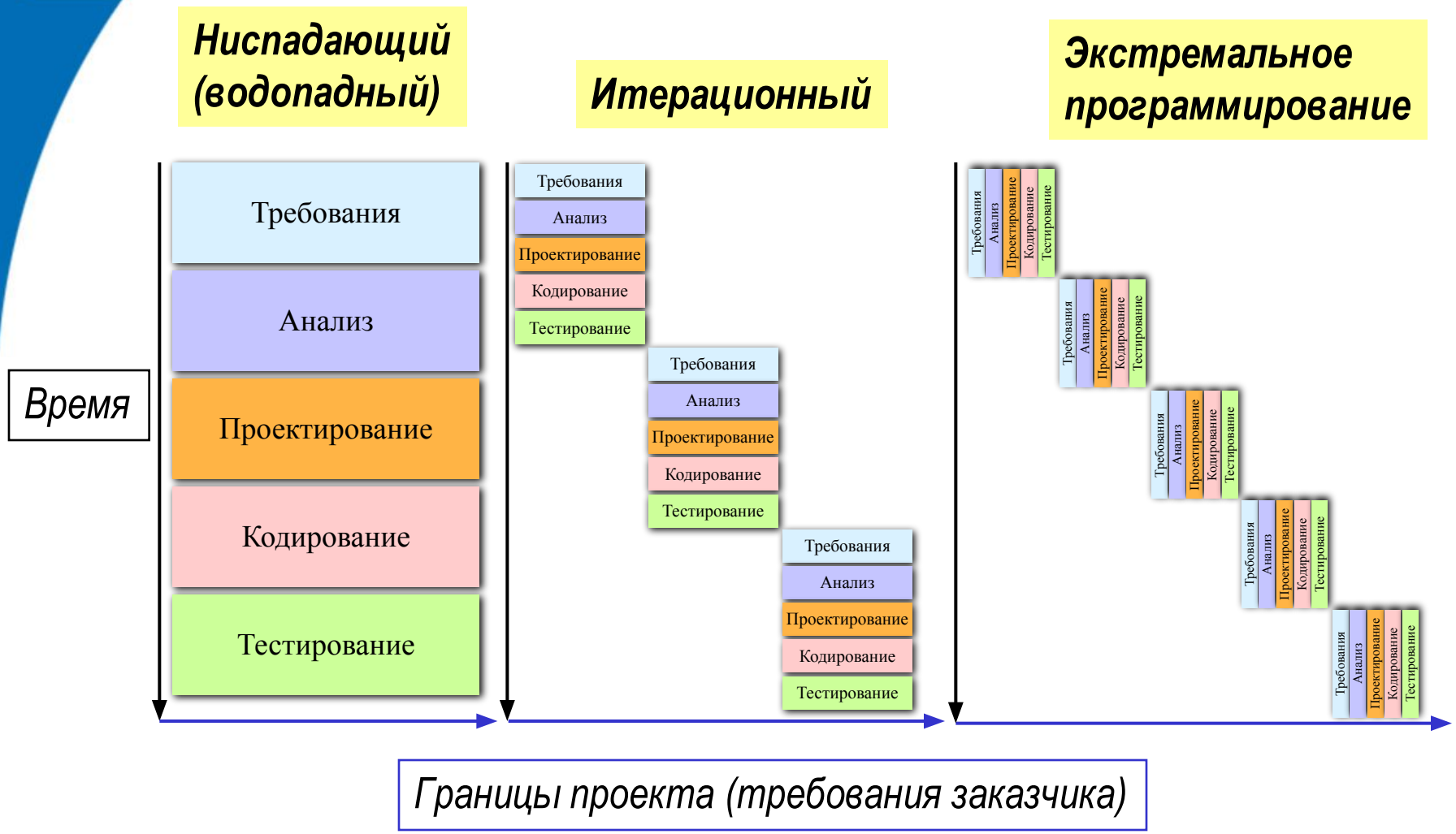
- **Нуждаются в команде разработчиков**
- **Никто полностью не знает всех их аспектов**
- **Имеют длительное время жизни (> 10 лет)**
- **НЕВОЗМОЖНО переписать их заново ради**
  - Исправления ошибок или расширения возможностей
  - Переноса на другую платформу
- **Требуют организованности, дисциплины и выбора правильных инструментов**

Фазы разработки ПО

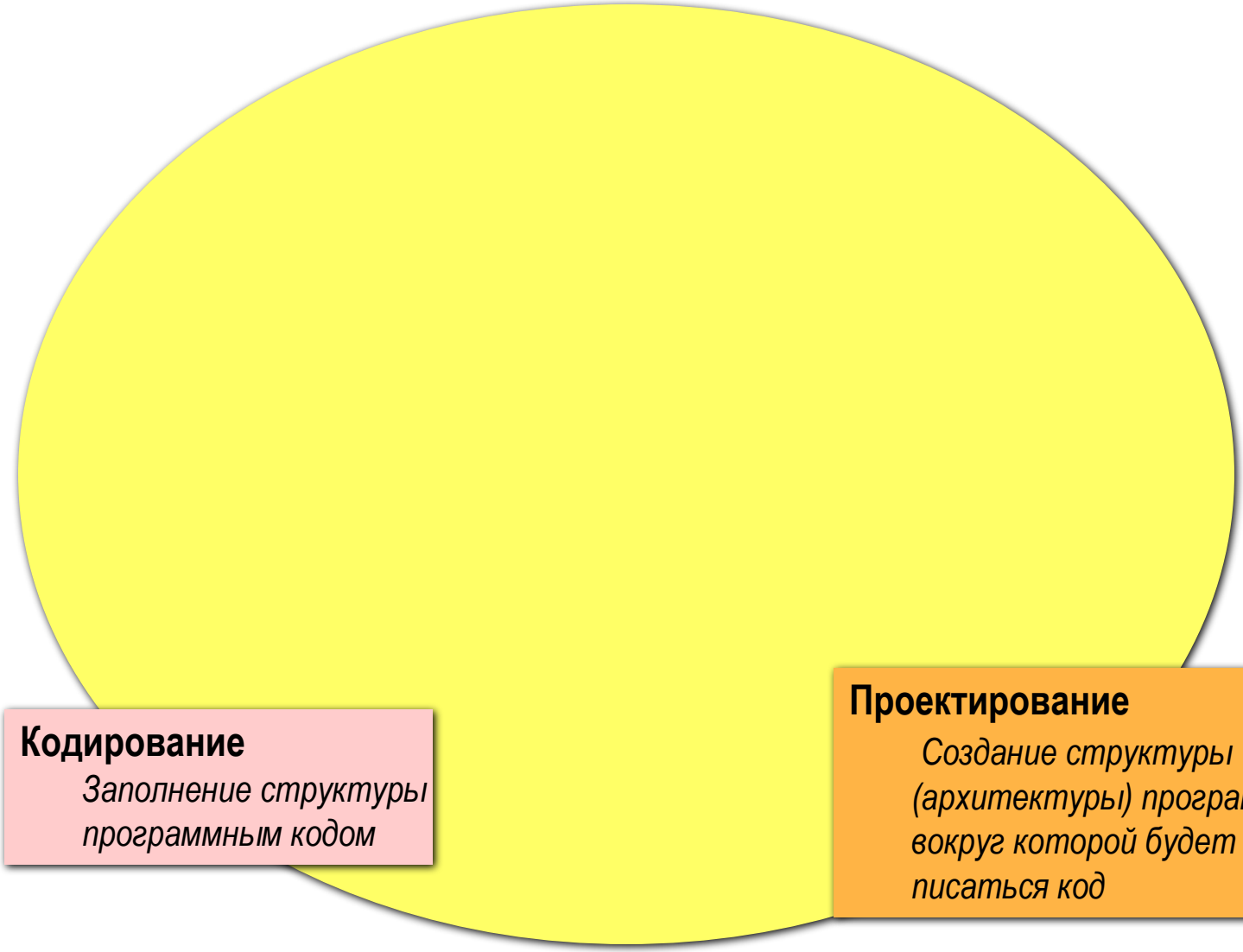
Процессы создания ПО

- **Процесс создания ПО это**
  - Набор действий (таких как сбор требований, анализ, проектирование, кодирование, тестирование) объединённых и расположенных в определённом порядке согласно выбранному стилю производства программного обеспечения
- **Последние тенденции: Динамический процесс создания программного обеспечения**
  - Требования заказчика со временем меняются (эволюционируют)
  - Заказчик, удовлетворенный в момент сдачи проекта значит больше, чем жесткое следование его (заказчика) первоначальным требованиям

Примеры стилей производства ПО



Фазы разработки ПО представленные в этом курсе



Кодирование

Заполнение структуры программным кодом

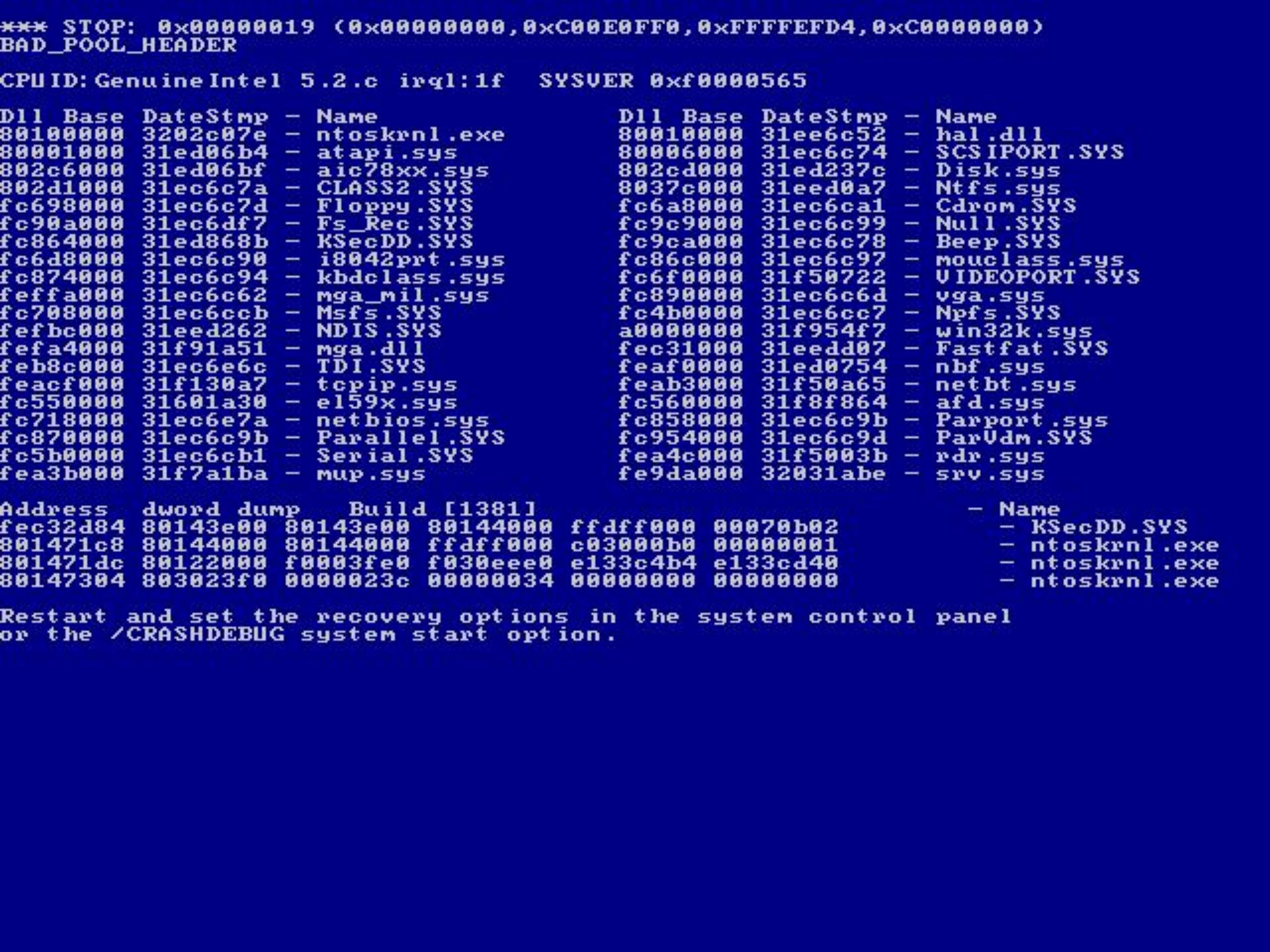
Проектирование

Создание структуры (архитектуры) программы, вокруг которой будет писаться код

Надежность программного обеспечения

Надежность программного обеспечения

Степень уверенности пользователя в том, что ПО будет работать ожидаемым образом, и без сбоев при использовании его в нормальных режимах работы


*** STOP: 0x00000019 (0x00000000,0xC00E0FF0,0xFFFFEFD4,0xC0000000)
BAD_POOL_HEADER

CPUID:GenuineIntel 5.2.c irql:1f SYSVER 0xf0000565

Dll Base DateStmp - Name
80100000 3202c07e - ntoskrnl.exe
80001000 31ed06b4 - atapi.sys
802c6000 31ed06bf - aic78xx.sys
802d1000 31ec6c7a - CLASS2.SYS
fc698000 31ec6c7d - Floppy.SYS
fc90a000 31ec6df7 - Fs_Rec.SYS
fc864000 31ed868b - KSecDD.SYS
fc6d8000 31ec6c90 - i8042prt.sys
fc874000 31ec6c94 - kbdclass.sys
feffa000 31ec6c62 - mga_mil.sys
fc708000 31ec6ccb - Msfs.SYS
fefbc000 31eed262 - NDIS.SYS
fefa4000 31f91a51 - mga.dll
feb8c000 31ec6e6c - TDI.SYS
feacf000 31f130a7 - tcpip.sys
fc550000 31601a30 - el59x.sys
fc718000 31ec6e7a - netbios.sys
fc870000 31ec6c9b - Parallel.SYS
fc5b0000 31ec6cb1 - Serial.SYS
fea3b000 31f7a1ba - mup.sys

Dll Base DateStmp - Name
80010000 31ee6c52 - hal.dll
80006000 31ec6c74 - SCSIPORT.SYS
802cd000 31ed237c - Disk.sys
8037c000 31eed0a7 - Ntfs.sys
fc6a8000 31ec6ca1 - Cdrom.SYS
fc9c9000 31ec6c99 - Null.SYS
fc9ca000 31ec6c78 - Beep.SYS
fc86c000 31ec6c97 - mouclass.sys
fc6f0000 31f50722 - VIDEOPORT.SYS
fc890000 31ec6c6d - vga.sys
fc4b0000 31ec6cc7 - Npfs.SYS
a0000000 31f954f7 - win32k.sys
fec31000 31eedd07 - Fastfat.SYS
feaf0000 31ed0754 - nbf.sys
feab3000 31f50a65 - netbt.sys
fc560000 31f8f864 - afd.sys
fc858000 31ec6c9b - Parport.sys
fc954000 31ec6c9d - ParVdm.SYS
fea4c000 31f5003b - rdr.sys
fe9da000 32031abe - srv.sys

Address dword dump Build [1381] - Name
fec32d84 80143e00 80143e00 80144000 ffdff000 00070b02 - KSecDD.SYS
801471c8 80144000 80144000 ffdff000 c03000b0 00000001 - ntoskrnl.exe
801471dc 80122000 f0003fe0 f030eee0 e133c4b4 e133cd40 - ntoskrnl.exe
80147304 803023f0 0000023c 00000034 00000000 00000000 - ntoskrnl.exe

Restart and set the recovery options in the system control panel
or the /CRASHDEBUG system start option.


Microsoft
intel inside
Ctrl
Alt
Delete

Синий Экран Смерти (СЭС)

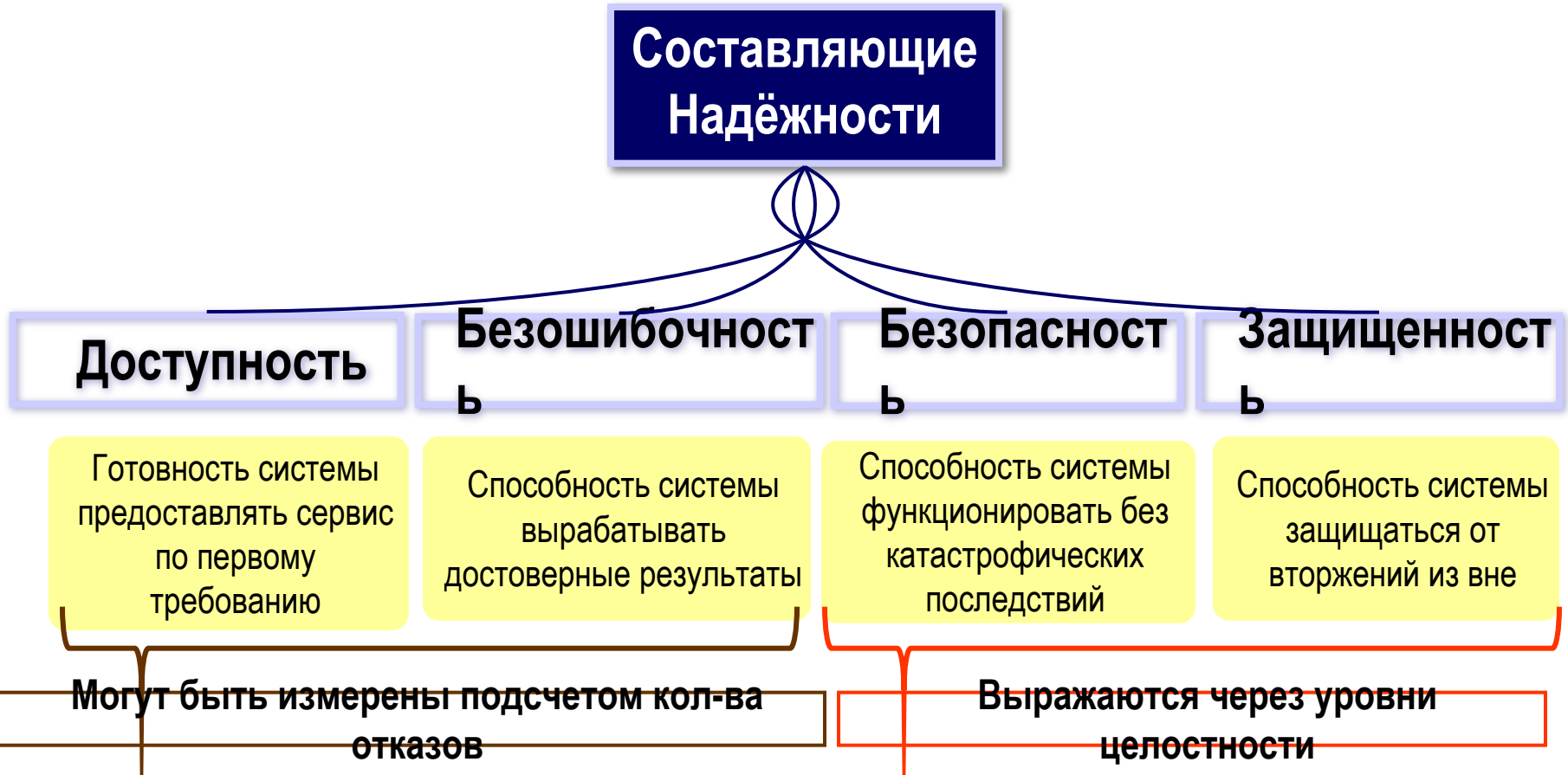
Еще пример СЭС в общественном месте. Удручающее зрелище, не так ли?

Значима ли надёжность программного обеспечения?

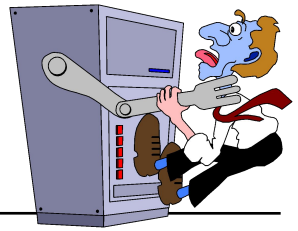
- **Несомненно значима! На маркетинговом уровне ☺**
  - Ни один поставщик не скажет, что его программное обеспечение ненадёжно
  - Ни одна команда разработчиков не сообщит, что разрабатывает ненадёжное ПО
- **В действительности, есть огромное количество ПО, ошибки в котором нас никак не задевают**
- **Не все программы нуждаются в том, что бы требование надёжности ставилось на первый план**
- **Сбой полезных, но некритичных программ все еще приемлемо 😐**
  - Если произойдет сбой во время этой презентации – достаточно просто перегрузить компьютер
  - Если Ваш текстовый редактор зависнет во время набора важного документа, это не принесет Вам ощутимого вреда, если Вы часто сохраняли результаты своей работы

Надёжность программного обеспечения

- **Надёжность ≠ Пригодность**
  - Пример: текстовый редактор



Предостережение относительно подсчета количества отказов



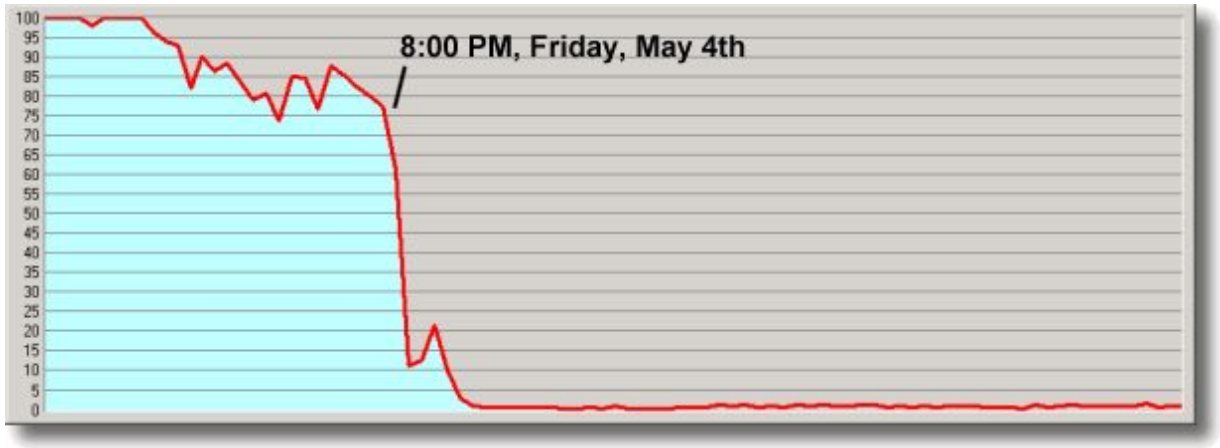
Приемлемо ли ПО с 99.9% защитой от отказа? Все зависит от ситуации... Ведь может случиться, что оставшийся 0.1% это:

- **1 документ в год потерянный в момент редактирования**
  - Хорошо ☺
- **2 происшествия в месяц в Международном Аэропорту в Лондоне**
  - ☹ ☹
- **22 000 чеков в час выписанные с ошибочных счетов по всей Америке**
  - ☹ ☹

Следствие: количество отказов ПО необходимо анализировать в контексте приложения

Сбои ПО: Доступность

- **Атаки отказа от обслуживания**
  - Пример: атака GRC.com
    - Атаковано 195 серверов Windows 2000 исполнявших недостаточно защищенную версию web-сервера Microsoft IIS. Для проникновения в систему хакерами была использована брешь в защите IIS. Это привело к остановке серверов и временной их недоступности



Сбои ПО: Безошибочность

- **Январь 15, 1990: на 9 часов остановлена общенациональная телефонная сеть США**
  - месяц ранее AT&T обновила ПО на 114 коммутируемых телефонных станциях
  - Причина: 1 неуместный оператор “break” в программе на языке C
- **Январь 2001: отзывается 230,000 единиц новых мобильных телефонов с доступом в Интернет**
  - Пользователи сообщают, что их телефоны зависают после посещения некоторых web-узлов, а после перезапуска телефона все сохраненная на нем информация (адреса, ссылки, записи) теряется
- **Matracom 6500 PABX (телефонный коммутатор)**
  - Искажение случайных телефонных разговоров
  - Внезапное прерывание длинных телефонных звонков
- **Windows NT**
  - Сентябрь 1997: повреждение силовой установки судна USS Yorktown
  - Причина: крах Windows NT 4.0

Сбои ПО: Безопасность

- **1986: Медицинская облучающая установка Therac 25 убила несколько пациентов**
  - Причина: недостаточно тестированное ПО установки
- **Июнь 4, 1996: 1-й полет ракеты Ariane 5 завершился неудачей: сработал механизм самоуничтожения**
  - Причина: проверенный временем код системы управления ракетой Ariane 4 был перенесен на Ariane 5, но не был протестирован.
- **2000: Большая автомобильная катастрофа на скоростном шоссе во Франции**
  - Причина: Неисправность ПО тормозной системы автомобиля. Производитель автомобиля признал свою ответственность за случившееся.

Сбои ПО: Защищенность

- **Ноябрь 2, 1988 Интернет-червь**
  - Самораспространяемая программа начала свое шествие через Интернет
  - Эта программа (червь) заражала компьютеры VAX и Sun работающие под Berkeley UNIX, и использовала их для атаки на другие компьютеры
  - За нескольких часов она распространилась на все Соединенные Штаты, инфицируя тысячи компьютеров и делая многие из них неработоспособными, чрезмерно нагружая их своим кодом
  - Причина: необнаружаемое переполнение буфера в функции gets() библиотеки времени выполнения языка C
- **Множество занимательных историй о вирусах, в особенности для ОС Windows**

… и 30% проектов ПО, которые не дожили даже до этих стадий

- **Модернизация налогового управления США**
  - $4 миллиарда, прекращена в начале 1997
- **Система анализа отпечатков пальцев для ФБР**
  - $500 миллионов, прекращена
- **Bell Atlantic 411**
  - Ноябрь 1996, устарела, принято решение систему не модернизировать

Критичность программного обеспечения

ПО и Критичность

- **Критичность по отношению к бизнес-процессам**
  - Сбой программного обеспечения может привести к значительным финансовым потерям и даже к полной остановке бизнеса
  - Например, система межбанковских платежей

- **Критичность по отношению к решаемой задаче**
  - Сбой программного обеспечения может привести к невыполнимости поставленной задачи
  - Например, спутник для исследования Марса

- **Критичность по отношению к безопасности**
  - Сбой программного обеспечения может привести к человеческим жертвам или большим разрушениям
  - Например, самолет

Стандарты на критичное к безопасности ПО

- **RTCA/EUROCAE DO-178B**
  - Международный стандарт на критичное для безопасности ПО в области авиастроения
- **IEC 880**
  - Стандарт на ПО для атомных электростанций
- **IEC61508 / DEF STAN 00-55/56**
  - Европейский стандарт безопасности
- **Руководство разработчика ПО для транспортных средств**
  - Стандарт безопасности, предложенный Ассоциацией разработчиков безотказного ПО для автомобильной промышленности MISRA (Motor Industry Software Reliability Association)

Уровни критичности ПО согласно DO-178B

Уровень критичности	Последствия от ошибки/сбоя ПО
Уровень A	**Катастрофические** *(Продукты уровня A сообщают экипажу самолета о его положении в пространстве и предотвращают его от падения, н. п. системы управления полетом, авиационные картографические базы, некоторые дисплеи)*
Уровень B	**Опасные/Значительные** *(Системы уровня B: слежение за движением и уклонение от столкновений)*
Уровень C	**Большие** *(Системы уровня C: связь и управление каналами связи)*
Уровень D	**Незначительные** *(Системы уровня D: системы обеспечения комфорта)*
Уровень E	**Без последствий** *(Системы уровня E: развлекательные системы)*

IEC61508 Уровни безопасности-сложности-целостности SCIL (Safety-Complexity-Integrity Levels)

Уровень SCIL	Последствия от ошибки/сбоя ПО
SCIL 4	Смерть одного или нескольких людей, существенные финансовые потери *(Область: аэрокосмическая, медицинские системы, системы управления движением, системы управления опасными процессами, системы торможения)*
SCIL 3	Серьезные телесные повреждения или финансовые потери *(Область: управление силовыми установками средств передвижения)*
SCIL 2	Неудобство или недовольство *(Область: кассовые терминалы в супермаркетах, аппараты выдачи сигарет/напитков)*
SCIL 1	Без последствий *(Область: студенческие проекты, исследования)*

Уровни целостности предложенные MISRA

Уровень целост-ности	Возможность контроля со стороны водителя	Оценка допустимости отказа	*Примеры возможных последствий при сбое ПО автомобиля*
4	Не поддается контролю	Абсолютно недопустимо	*Обесточивание усилителя рулевого управления*
3	Сложно контролируется	Чрезвычайно редко	*Отказ тормозной системы*
2	Утомляет	Редко	*Неработоспособность механизма очистки лобового стекла*
1	Отвлекает	Не желательно	*Неработоспособность стеклоподъемника*
0	Только вызывает неудобство	Допускается	*Неработоспособность радио/CD плеера*

Программное обеспечение и безопасность

Стандарты безопасности ПО

- **TCSEC (Оранжевая книга)**
  - Критерии оценки безопасности высоконадежной компьютерной системы
- **Общие критерии оценки безопасности в Информационных технологиях (ИТ) (ISO/IEC 15408-1)**
  - Критерии оценки безопасности ИТ
  - 7 уровней оценки безопасности

Уровни оценки безопасности (EALs)

EAL	Ограничения на разрабатываемое ПО
EAL7	Формально доказанная корректность + тестирование
EAL6	Использование доказательства корректности при проектировании + тестирование
EAL5	Проектирование с использованием формальных методов + тестирование
EAL4	Методологическая проектирование, тестирование и исправление
EAL3	Проведены методологические тесты и проверки
EAL2	Проведен структурный тест
EAL1	Оттестирована функциональность

Развитие программного обеспечения

Программное обеспечение нуждается в развитии

- **Исправление ошибок**
- **Перенос на новые архитектуры**
  - Программное обеспечение остается актуальным длительное время
    - Например, проблема 2000-го года
  - Наиболее успешное ПО переживает аппаратное обеспечение, для которого разрабатывалось
    - Например, VAX/VMS
  - Когда новое оборудование становится доступным, дешевле портировать существующие приложения, чем написать все с нуля
    - Например, Intel IA-64
- **Улучшения и новые возможности**
  - Например, Dos, Windows 3.1, Windows 95/98/ME, Windows NT/2000/XP, ...

Цена затрат на разработку ПО в США в 2001 году

- **Обычно производительность разработки ПО составляет:**
  - От 2 до 20 строк рабочего кода (LOC) в день на 1 программиста
- **Средняя стоимость услуг программиста в день** (включая все расходы):
  - От 150 до 500 USD/день
- **Средняя стоимость 1-й строки рабочего кода (LOC)**
  - От 10 до 50 USD
- **Стоимость разработки приложения в 100,000 LOC составляет**
  - В среднем от 1 до 5 миллионов USD

Развитие ПО – необходимость

- **Нельзя просто выбросить программу и переписать ее заново**
  - Во-первых – стоимость
  - Во-вторых – время и рынок. А это, как правило, значительно важнее!
- **Возможно, от разрабатываемой Вами программы не требуется сверхнадежность, но...**
- **… определенно, требуется способность развиваться**
  - В ногу со временем
  - При приемлемой стоимости изменений
- **Примеры**
  - Компилятор GNU Ada/C/C++ состоит из более чем 2 миллионов LOC
  - Редактор Emacs – приблизительно из 1.4 миллионов LOC
  - GNU/Linux – приблизительно из 4 миллионов LOC

Тенденции в разработке ПО

Все чаще и чаще разработка программного обеспечения касается не создания нового ПО, а расширения и модифицирования уже существующих систем

Выводы

В зависимости от Вашей предметной области

- **Возможно все или только некоторые параметры надежности разрабатываемого ПО имеют для Вас значение**
  - Доступность, безошибочность, безопасность, защищенность
- **… но почти во всех случаях**

Основой является способность ПО развиваться