

# Лекция 10: Сопровождение и реинжиниринг

Юрий Литвинов  
y.litvinov@spbu.ru

07.05.2024

# Сопровождение

- ▶ Сопровождение неизбежно:
  - ▶ Развитие бизнес-процессов
  - ▶ Изменение внешнего окружения
  - ▶ Исправление ошибок
  - ▶ Повышение производительности
- ▶ Организации зависят от ПО
  - ▶ Иногда критически
- ▶ Сопровождение стоит денег и усилий

# Законы Лемана

- ▶ Непрерывное изменение
- ▶ Увеличение сложности
- ▶ Саморегулирование
- ▶ Сохранение организационной стабильности
- ▶ Сохранение осведомлённости
- ▶ Ухудшение качества
- ▶ Система обратной связи

# Унаследованные (legacy) системы

- ▶ Жизненный цикл — 20 лет и более
- ▶ Высокие риски при замене
  - ▶ Нет технического описания
  - ▶ Система переплетена с бизнес-процессами
  - ▶ Система является единственным источником знаний о некоторых бизнес-правилах
    - ▶ Включая ошибки системы!
  - ▶ Риски разработки новой системы

# Стоимость поддержки

- ▶ Разные команды → разный стиль
- ▶ Устаревшие технологии → сложно искать кадры
- ▶ Качество и актуальность документации
  - ▶ Иногда её просто нет
  - ▶ Иногда нет даже кода
- ▶ Архитектурная эрозия
- ▶ Оптимизации
- ▶ Дублирование и неконсистентность данных

# Что делать?



# Модернизация программного обеспечения

- ▶ Учитывать важность для бизнеса
- ▶ Учитывать качество
- ▶ Учитывать аппаратное обеспечение и окружение
- ▶ К разным частям системы могут применяться разные стратегии
  - ▶ К разным программам в составе системы — тем более

# Сопровождение

- ▶ Исправление ошибок
- ▶ Адаптация к условиям эксплуатации
- ▶ Изменение функциональности
- ▶ Профилактическое сопровождение



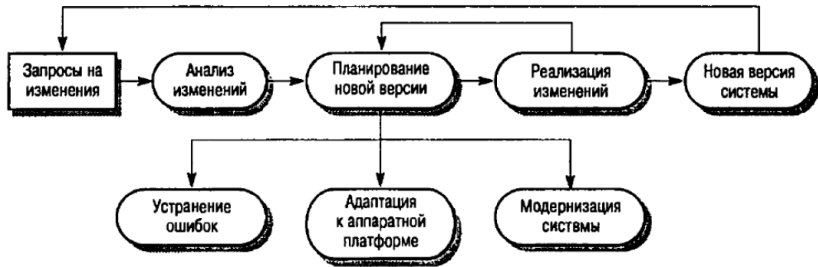
# Сопровождение

- ▶ Исправление ошибок
  - ▶ Адаптация к условиям эксплуатации
  - ▶ Изменение функциональности
  - ▶ Профилактическое сопровождение
- 
- ▶ 65% — выполнение новых требований
  - ▶ 18% — адаптация к новому окружению
  - ▶ 17% — исправление ошибок

# Факторы стоимости сопровождения

- ▶ Стабильность команды разработчиков
- ▶ Ответственность согласно контракту
  - ▶ Оригинальные разработчики не мотивированы облегчить сопровождение
- ▶ Квалификация специалистов
- ▶ Возраст и структура программы

# Процесс сопровождения



# Нарушения процесса

- ▶ Часто бывает нужно
  - ▶ Критическая ошибка в системе
  - ▶ Изменение рабочего окружения
  - ▶ Неожиданные изменения бизнеса
    - ▶ Например, изменения законодательства
- ▶ Потеря целостности требований и архитектуры
- ▶ Выбор быстрого решения, а не правильного
  - ▶ Откатить хотфикс и «сделать нормально»

# Прогнозирование сопровождения

- ▶ Количество и сложность интерфейсов
- ▶ Количество изменяемых системных требований
- ▶ Бизнес-процессы, в которых используется данная система
- ▶ Взаимосвязанность и сложность компонентов
  - ▶ Вспомним лекцию про метрики

# Оценка удобства сопровождения

- ▶ Количество запросов на корректировку системы
- ▶ Количество корректировок, которые затронули каждый модуль
- ▶ Среднее время, потраченное на анализ причин системных сбоев и отказов
- ▶ Среднее время, необходимое на реализацию изменений
- ▶ Количество незавершенных запросов на изменения

# Личные качества сопровождающего программиста

- ▶ Гибкость в работе
- ▶ Творческий подход к задачам
- ▶ Широкий профессиональный кругозор
- ▶ Хорошая память
- ▶ Терпение
- ▶ Самостоятельность
- ▶ Ответственность и самокритичность

# Техподдержка, виды контрактов

- ▶ Фиксированный объём работ
- ▶ Техподдержка на определённый срок
- ▶ Поддержка по необходимости (Time and Materials)
- ▶ Сопровождение продукта



# Линии поддержки

- ▶ Линия 1 — сбор информации, решение проблем по FAQ
  - ▶ Неквалифицированные кадры, не решают технические проблемы
- ▶ Линия 2 — помощь линии 1, решение известным способом
  - ▶ Специалисты, разбирающиеся в продукте
  - ▶ Некоторая техническая работа, типа правки данных
  - ▶ Может быть несколько специализированных групп
- ▶ Линия 3 — решение неизвестных проблем
  - ▶ Настоящая команда сопровождения/разработки

# Улучшение

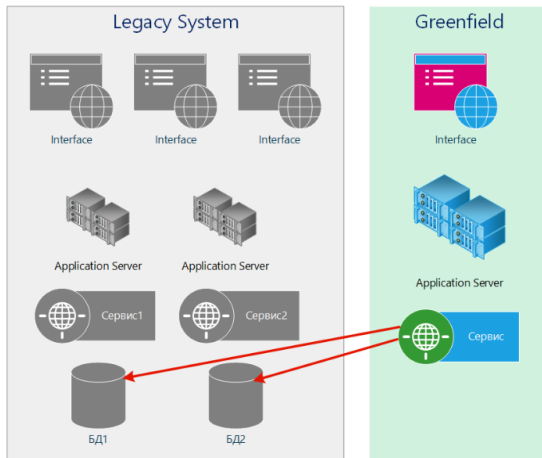
## Как работать с унаследованным кодом

- ▶ Понимать, что код уже приносит прибыль
  - ▶ Каким бы плохим он ни был, он лучше ненаписанного
  - ▶ Ответственность
- ▶ Reverse engineering
  - ▶ Восстановление архитектуры
  - ▶ Отслеживание цепочек вызовов
  - ▶ Исследовательская отладка
  - ▶ Промышленная археология
  - ▶ Документирование результатов

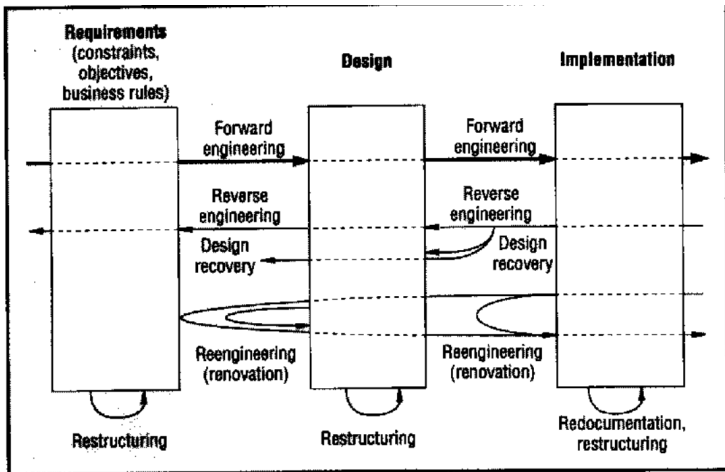
# Советы по процессу

- ▶ Не переписывайте код
- ▶ Не меняйте технологии/парадигму
  - ▶ Но есть распределённые приложения
- ▶ Помните о бизнес-интересах
- ▶ Не забывайте про логирование
- ▶ Не забывайте про тестирование
  - ▶ Characterization testing
- ▶ Постройте чёткий процесс релизов
- ▶ Определите стратегию версионирования кода
- ▶ Контролируйте качество кода
  - ▶ Code review
- ▶ Выделяйте (и переписывайте) отдельные модули

# «Приложение-душитель»



# Реинжиниринг



© E. Chikofsky et al. Reverse engineering and design recovery: a taxonomy

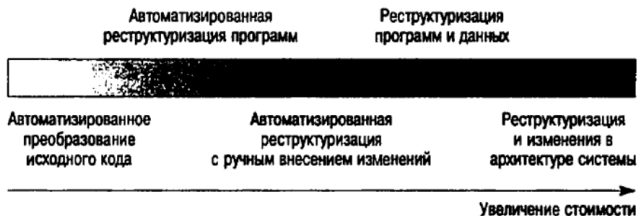
# Реинжиниринг против полного переписывания

- ▶ Унаследованного кода очень много
  - ▶ 120 млрд строк на 1990 год, и это было только начало
- ▶ Снижение рисков
- ▶ Снижение затрат
  - ▶ Примерно в четыре раза дешевле, чем разработка с нуля
  - ▶ Автоматизируем
- ▶ Ограничен в возможностях улучшения системы
  - ▶ Только частично решает проблему сопровождения

# Реинжиниринг против полного переписывания



# Стоимость реинжиниринга





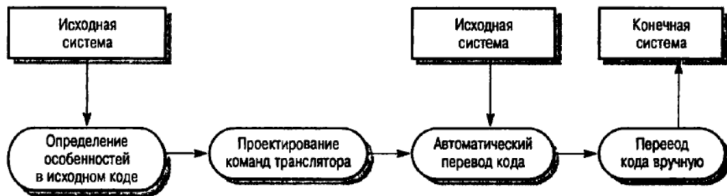
## Факторы стоимости

- ▶ Качество программного обеспечения, которое подвергается реинжинирингу
- ▶ Наличие средств поддержки процесса реинжиниринга
- ▶ Объем необходимого преобразования данных
- ▶ Наличие необходимых специалистов

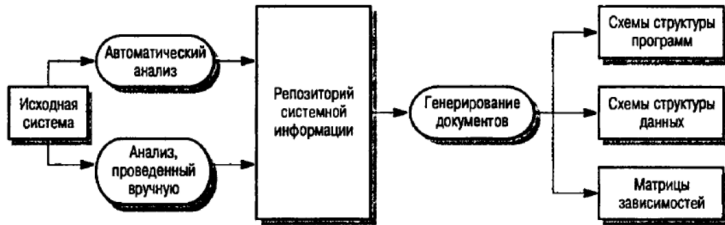
# Процесс реинжиниринга



# Перевод исходного кода



# Анализ программ



# Модификация структуры программ



# Автоматизация

- ▶ Инвентаризация исходного кода
- ▶ Разбор исходного кода
- ▶ Выделение графов потока управления, потока данных
- ▶ Анализ потока управления, потока данных
- ▶ Удаление мёртвого кода, извлечение бизнес-правил
- ▶ Генерация АСД целевого языка
  - ▶ Переписыватели деревьев, например, Stratego/XT
- ▶ Генерация кода и правил сборки

# Автоматизация, проблемы

- ▶ Потеря комментариев
  - ▶ Зависит от используемого инструмента
- ▶ Утрата связи с документацией
  - ▶ Скорее всего, она всё равно устарела
- ▶ Жесткие требования к компьютерной технике
  - ▶ Зависит от используемого инструмента

# Разбиение на модули

- ▶ Реинжинирить только нужное:
  - ▶ Интенсивность сбоев
  - ▶ Частота изменений
  - ▶ Сложность
    - ▶ Метрики!
- ▶ Явное вынесение модулей
  - ▶ Абстракции данных
  - ▶ Аппаратные модули
  - ▶ Функциональные модули
  - ▶ Модули поддержки отдельных процессов
- ▶ Делается вручную



# Изменение данных

- ▶ Критично для информационных систем
- ▶ Причины изменений:
  - ▶ Нарушение данных
    - ▶ Дублирование и неконсистентность
    - ▶ Долгие сроки хранения и устаревание
  - ▶ Программные ограничения
    - ▶ Кто помнит телефоны «не более 100 SMS»?
  - ▶ Эволюция системной архитектуры
    - ▶ Распределённые системы
- ▶ Требуется анализ кода на литералы

## Модификация структуры программ

