



**Return
on
Intelligence**

Core Systems Transformation Solutions



**Достаточно хорошее
программное обеспечение.
Где остановиться?**



October 24, 2013

Damir Tennishev, Project manager, PhD

Время

единственный невозполнимый ресурс



Пример 1

Задача

На приставке с ограниченным объёмом оперативной памяти и жёсткими требованиями к производительности требуется вести протокол работы программы для анализа ошибочных ситуаций.

Решение

Протокол ведётся в циклическом буфере памяти размером в несколько сотен килобайт. Как выяснилось – важнее всего иметь последние перед ошибкой строки протокола.

Пример 2

Задача

Система управления технологическим объектом должна быть показана на выставке в реальной работе. При этом надежность системы достаточно низкая (большое число ошибок).

Решение

1. Написано устойчивое ядро, запускающее остальные компоненты
2. В случае ошибки в компоненте, ядро делает скриншот экрана и перегружает компонент
3. Все устройства ввода-вывода, работающие с внешними приборами, сохраняют значение сигнала на момент зависания системы

Пример 3

Задача

Производительность крупной системы после выхода в производство была признана неприемлемой. Все операции выполнялись крайне медленно. Увеличение мощности оборудования не привело к желаемым результатам. Необходимо обеспечить требуемую производительность.

Решение

1. Был выявлен основной набор операций, которые использовались клиентом.
2. Была произведена оптимизация производительности только для наиболее часто используемых операций.
3. Все остальные операции были оставлены без изменения.

Пример 4

Задача

Интегрированная среда разработки предметно-ориентированного языка не справлялась с генерируемым кодом размером более 5 мегабайт.

Попытка решения

Код редактора был переработан с целью поддержки виртуального окна в исходном коде.

Решение

1. От генерации кода было решено отказаться.
2. Вместо собственной интегрированной среды было решено использовать MS Visual Studio IDE

Пример 5

Задача

В крупном игровом проекте было решено использовать исключения (exceptions), поскольку это естественный и удобный механизм обработки ошибок в языке C++.

Последствия

- Объём кода увеличился на 30% только за счёт включения механизма обработки исключений.
- Производительность упала вследствие увеличения объёма кода.
- При переносе игры на приставку выяснилось, что компилятор не поддерживает механизм исключений.

Почему совершенство недостижимо

- Условия меняются быстрее, чем реализуется проект
 - Понимание системы заказчиком и исполнителем
 - Внешние обстоятельства (оборудование, окружение)
 - Не надо завершать на раннем этапе
- У всего нового есть период отторжения
 - При итеративной разработке заказчик не успевает привыкнуть к новой версии системы
- Сложная система, построенная с нуля, никогда не заработает
 - Всегда надо начинать с работающей простой системы
- Методы из других дисциплин работают плохо
 - Мы работаем почти «вслепую»
- Не думайте, что другие знают лучше вас
 - Чаще всего заказчик знает о задаче меньше вас
 - Отстаивайте своё мнение

Что происходит?

Проект – система управления с обратной связью.

Звено	Проявление
Запаздывания	Система реагирует позже по времени или топологии
Интегральное	Система приобретает значительную инертность в управлении
Дифференциальное	Система сильно реагирует на локальные раздражители

Устойчивость и управляемость находятся на разных полюсах.

Результат

Если не понимать динамики системы, неизбежно перерегулирование.

Эффект второй системы или эффект маятника

Инвестиции увеличиваются в те элементы, которые вызывали проблемы в предыдущем проекте. Часто в те элементы, где проблемы проявлялись.

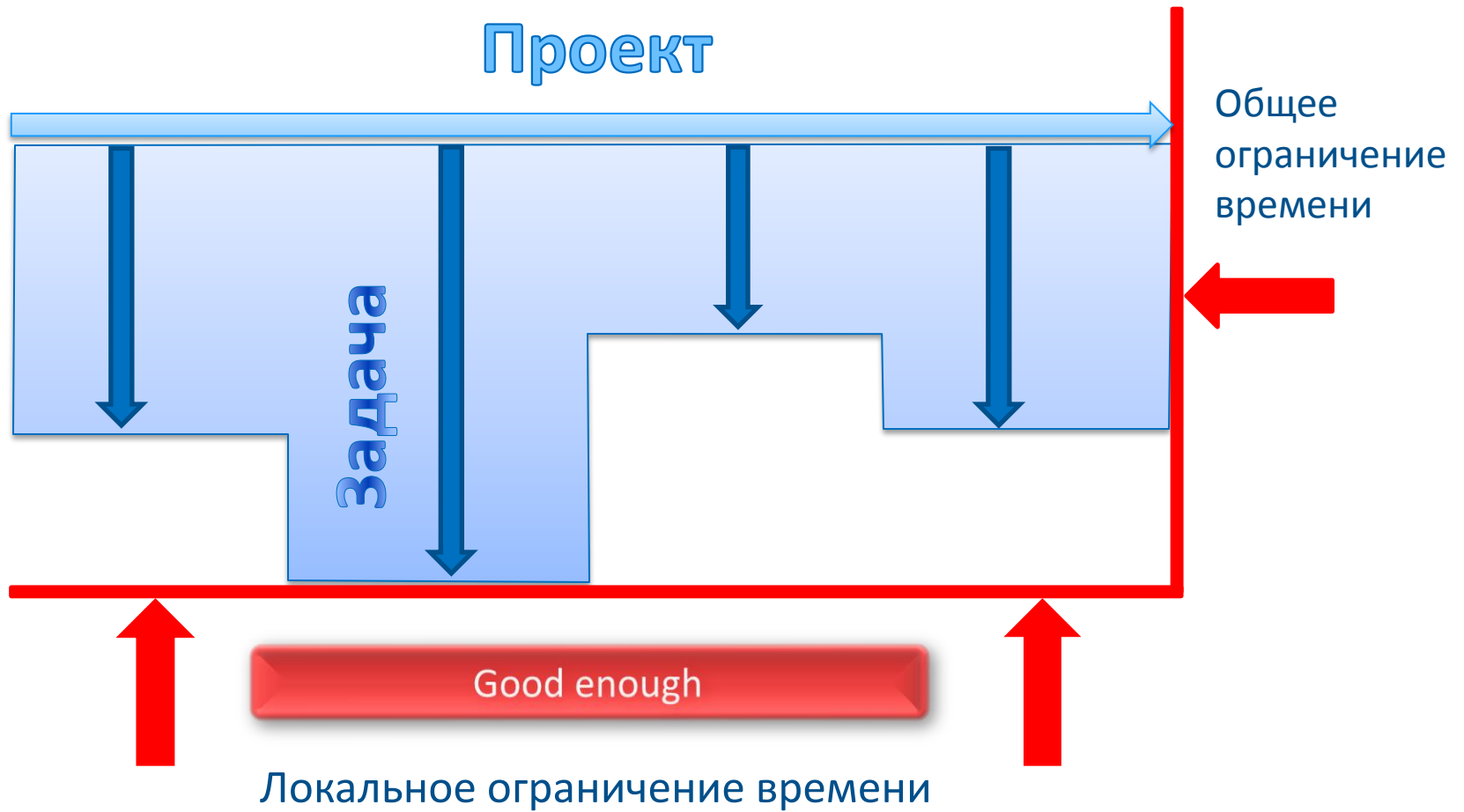
А нужно инвестировать в то, что работает в проекте.

Единственная разумная мысль в докладе

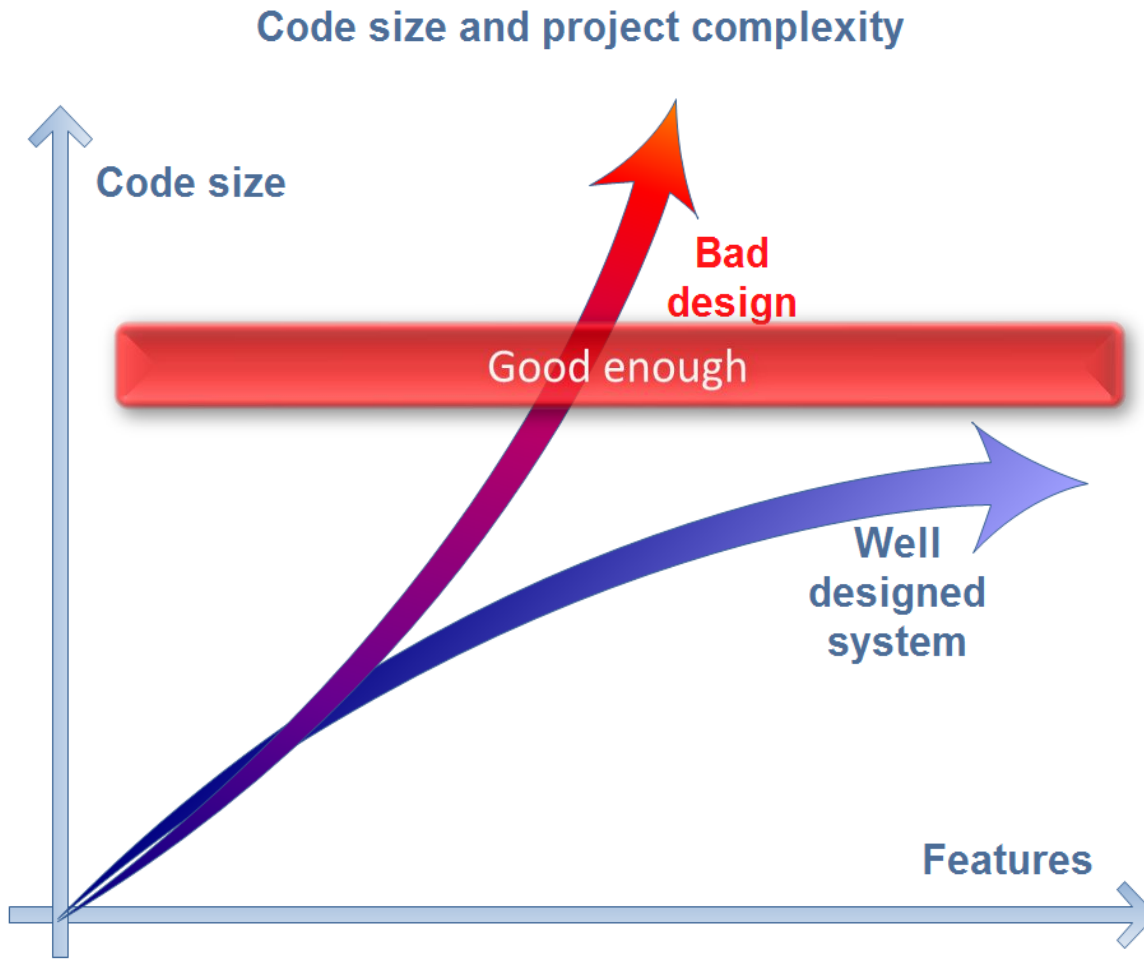
- Всегда имейте более важное или более интересное дело, чем текущая цель. Тогда вы будете её добиваться быстро и эффективно.
 - Становится жаль времени на эту задачу
 - Возможные ошибки воспринимаются не так остро
- Люди должны знать, что текущая задача – только небольшой этап. После завершения проекта есть следующий шаг. Намного более масштабный.
 - Дайте усилиям распространяться в ширину, а не только в глубину
 - Пустота после проекта вызывает торможение перед пропастью
- Ищите не ответственных, а заинтересованных
 - Тех, кто будет эту задачу любить
- Смотри на цель, и ты отбросишь всё лишнее
 - Фокус – это не умение концентрироваться на одной вещи, а умение отказываться от многих других.

Составляющие успеха

- Видеть систему в целом
- Критерий готовности (definition of done)
- Ограничивать время
 - Разделять на подзадачи с ограниченным временем (WBS)
 - Понимание куда и как может быть использовано выигранное время
- Начинать сложные задачи сейчас
 - Появятся понимание, вопросы, оценки
 - Проще продолжать, делегировать
 - Может оказаться простым
- Понимать приоритеты
 - Различать приоритет (priority) и уровень (severity) проблемы
- Принимать быстрые решения
- Различать ориентацию на процесс и на результат



Размер системы



- Код надо поддерживать
 - **Стоимость поддержки растёт незаметно в ходе проекта**
 - 100К строк на разработчика
 - Или 5 выделенных разработчиков на миллион строк кода
-
- Измерять пропорцию объёма кода к новой функциональности
 - Ставить ограничения на объём кода

Правильные вопросы

- **Нам** нужен этот проект?
- Я согласен с оценкой. Что можно сделать, чтобы реализовать в два раза быстрее или дешевле?
- Как сократить объём артефактов (код, документация, используемые технологии) в два раза?
- Как это решение связано с главными целями проекта?
- Что будет, если этого не сделать?
- Какие ещё есть варианты достижения желаемого результата?



Спасибо за вопросы



Return
on
Intelligence

Damir Tenishev

PhD, Project manager

Pulkovskoye shosse 40/4
St. Petersburg, 196158, Russia

+7(812) 383.5800 x. 3247
Damir.Tenishev@returnonintelligence.com

Не существует окончательных решений...