



Коммуникация нефункциональных требований в небольших проектах по заказной разработке ПО

Калугин Александр, PhD, PMP

Mercury Development
Project Director



*We should forget about small efficiencies,
say about 97% of the time: premature
optimization is the root of all evil.
Yet we should not pass up our opportunities
in that critical 3...*

*Нам следует забывать о небольшой
эффективности, например, в 97% случаев:
преждевременная оптимизация —
корень всех зол. Хотя мы не должны
отказываться от своих возможностей
в этих критических 3%*

Д. Кнут

Вводная информация:

- Разработка новой системы или существенная переработка – не доработка.
- Уже собраны аналитиком (или предоставлены заказчиком) функциональные требования к системе: use-cases, workflows
- Fixed-price, одна итерация, которая должна быть выпущена и иметь возможность продолжения



Цель выявления требований:

- Адекватная конкурентная оценка стоимости проекта – удовлетворенность подрядчика.
- Удовлетворенность заказчика результатами разработки
- Отсутствие препятствий успеху и дальнейшему развитию проекта, обусловленных неправильной архитектурой.



**Насколько достижение обозначенных целей
зависит от того насколько качественно
выявлены нефункциональные
требования?**



It depends...

Бизнес-цель проекта



- Жесткие ограничения (регуляция, жесткое реальное время, hardware ограничения целевой платформы)
- Конкурентные преимущества (продукт должен быть лучше чем продукт конкурентов)
- Не хуже чем есть (проекты портирования)

И

- «Пожелания» для новых продуктов

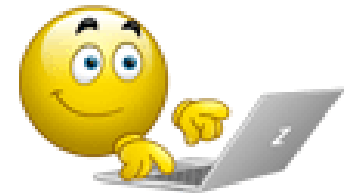
- Жесткие ограничения (регуляция, жесткое реальное время, hardware ограничения целевой платформы)

Не требуют выявления

(явно коммуницируются заказчиком как часть бизнес-цели).

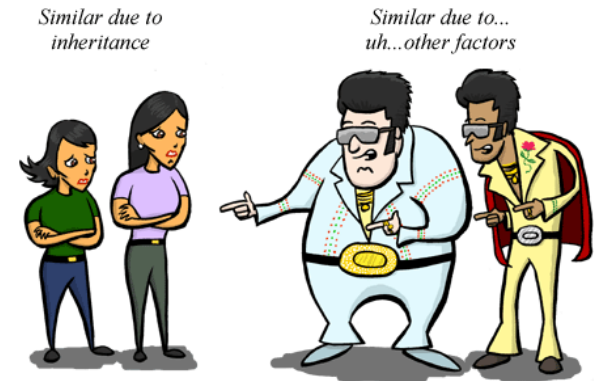
Примеры:

- Аудио-кодек, работающий в реальном времени
- AFP клиент, соответствующий стандарту и т. д.
- Выполнение на существующих рабочих станциях определенной минимальной производительности



- Конкурентные преимущества (продукт должен быть лучше чем продукт конкурентов)
- Не хуже чем есть (проекты портирования)

Источником нефункциональных требований является прототип/аналог



Метод выявления: reverse-engineering (например, нагрузочное тестирование аналога).

Коммуникация: «Замерили аналоги, которые Вы указали, получили вот такие результаты...» – нет необходимости объяснять значение отдельных замеров.

Проблемы: не всё можно измерить.

- «Пожелания» для новых продуктов

**Невыполнение конкретных
нефункциональных «пожеланий»
маловероятно будет причиной «неуспеха»
проекта...**



- «Пожелания» для новых продуктов



Если архитектура допускает оптимизацию в последующих версиях – то, даже если не достигнуты определенные показатели в первой версии, в последующих версиях ситуацию можно исправить

Цель: Корректные приоритеты различных противоречащих нефункциональных требований – иначе неадекватные оценки и неправильная архитектура

**Значение критерия А не должно
быть выше/ниже значения Х**

А что будет если $X+1$, $X+2$, это конец света?



Недостатки таких критериев:

1. Непонятны для заказчика
2. Представляют собой серьезный риск для подрядчика, обладая в то же время низкой бизнес-ценностью.
3. Может быть сложно реализовать, если не является прямым следствием архитектуры (пример – быстродействие или объем потребляемой памяти).
4. Может быть сложно протестировать Если не является прямым следствием архитектуры (система должна обеспечивать бесперебойную работу в течение 48 часов, в случае возникновения ошибки должна отсылать соответствующий SNMP trap)

- Определение границ проекта
 - Окружение
 - Вспомогательная функциональность
 - Отказ от требований
- Обеспечение правильной архитектуры
 - Список приоритетов
 - Дихотомии
 - Работа с функциональными требованиями



Цель: Выяснить ограничения проекта, чтобы отделить основное business-value от вспомогательного функционала:

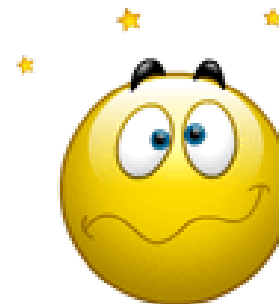
Ограничения	Предложенное значение	Вспомогательная информация
Поддерживаемые браузеры	Firefox 4.x, Safari 4.x	Информация о доле рынка
Поддерживаемые ОС	Mac OS C Leopard/Snow Leopard	
Совместимые устройства	iPhone 3GS, iPhone 4G, iPad, iPad 2, iPod 4G	

Другие примеры: поддерживаемые локализации, разрешения экрана, ориентация экрана и т.д.

Цель: Выяснить ограничения проекта, чтобы отделить основное business-value от вспомогательного функционала (примеры, сложность реализации)

- Инсталлятор/анинсталлятор
- Автоматическое обновление
- Пользовательская документация
- Подписывание бинарных файлов
- Запуск с RO носителей
- Voice over
- Help/Titles
- Large Fonts
- Scripting/Command-Line Interface
- Совместимость со стандартными форматами файлов
- Реклама
- Аналитика
- Интеграция с социальными сетями
- Dockable элементы и т. д.

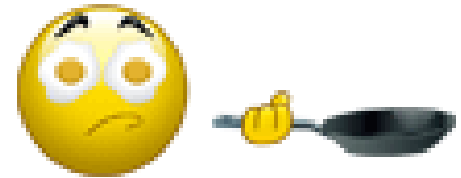
Сложность: Y



Цель: Избежать превращения пожеланий в ограничения. Коммуницируем с заказчиком предположения в виде:

1. Нет других требований к шифрованию/дешифрованию пользовательских данных
2. Нет явных требований к поведению UI контролов – при портировании
3. Нет явных требований по количеству обрабатываемых запросов/объеме пользовательских данных. Система должна обеспечивать корректную стабильную работу без потерь пользовательских данных.

и т.д.



Цель: выяснить относительные приоритеты различных аспектов работы системы.

Критерий	Рейтинг
Удобный, интуитивно понятный пользовательский интерфейс, время отклика.	
Защищенность пользовательских данных	
Дизайн/Красивый пользовательский интерфейс	
Производительность	
Расширяемость	
Отказоустойчивость	
Сохранность пользовательских данных	
Сохранение общей базы кода (при портировании) и т.д.	

Цель: Более четко интерпретировать отдельные нефункциональные требования

Примеры: 1. Производительность (драйвер дискового устройства)

Быстродействие ⇔ Сохранность от сбоев

Последовательный ⇔ Случайный доступ

Большой объем передаваемых данных ⇔ Задержка

2. Сетевые коммуникации

Гарантированность доставки ⇔ Низкие задержки

Защищенность ⇔ Производительность

Оптимальность трафика ⇔ Стандартное API

3. Веб-решение

Динамический пользовательский интерфейс ⇔ Кросс-браузерность

Простота поддержки, гибкость ⇔ Производительность/Расширяемость



Цель: Устранить из функциональных требований ограничения архитектуры, препятствующие выполнению нефункциональных требований.

Чеклист: Проверить соответствие имеющейся информации вновь выявленной. Просто «да/нет» выявленным нефункциональным критериям:

- Соответствует ли цели проекта
- Стабильность
- Производительность
- Безопасность...



Цель: Проанализировать и *модифицировать* use-case в случае если workflow содержит требования к объемам обрабатываемых данных.

- Если пользовательский интерфейс содержит списки, таблицы, иерархические структуры данных – возможны ли изменения в workflow, которые позволят загружать данные лишь частично ('50 more', paging, и т.д.)
- Возможно ли дополнительно ограничить объем данных путем введения дополнительной фильтрации?
- Подразумевается ли progress indicator для различных операций? – стоит ли их добавить?
- и т.д.



Цель: Проанализировать и *модифицировать* use-case в случае если workflow подразумевает возможность потери данных/перехода системы в некорректное состояние

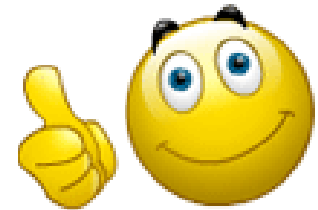
- Определена ли реакция системы, в случае если данные не могут быть обработаны корректно на каждом из этапов use-case-a – если нет -- доопределить.
- Возможно ли копирование и восстановление состояния системы без дополнительных действий – если да, добавить такую возможность.
- Требуется ли сохранение данных на жесткий диск на промежуточных этапах? Можно ли от этого отказаться?



Цель: Проанализировать и *модифицировать* use-case в случае если workflow содержит ограничения по производительности.

- Содержит ли workflow синхронные операции (описание чтение после записи)? Возможно ли заменить фоновыми/асинхронными операциями
- Подразумевается ли ветвление в зависимости от успешности результата предыдущей операции, возможно ли избежать ветвления?
- Возможно ли уменьшить количество обменов между приложением и back-end компонентами?

и т.д.

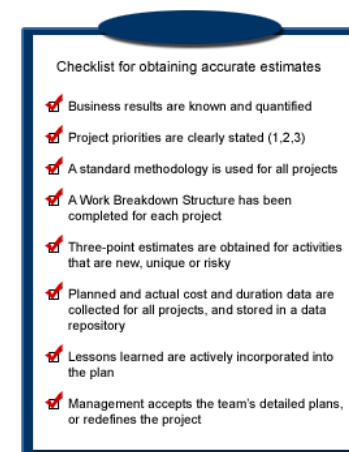


Что для этого нужно?

Достоинство нефункциональных требований – одинаковые для большинства проектов определенного типа.

Для успешной реализации необходимо:

1. Чеклист для ограничений.
2. Чеклист вспомогательной функциональности с примерами определенных особенностей.
3. Дихотомии для различных типов проектов.
4. Описание архитектурных преобразований для удовлетворения определенных нефункциональных требований.





Спасибо за внимание!

Ваши вопросы?

Калугин Александр

alex.kalouguine@gmail.com

<http://pmarcor.com/>

Все использованные графические ресурсы –
собственность правообладателей

