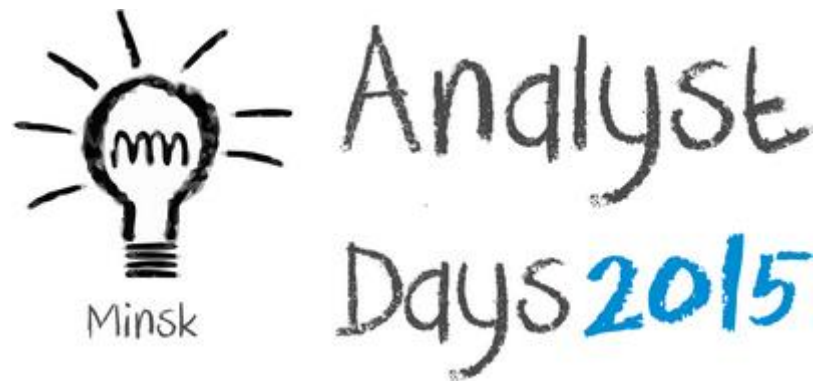
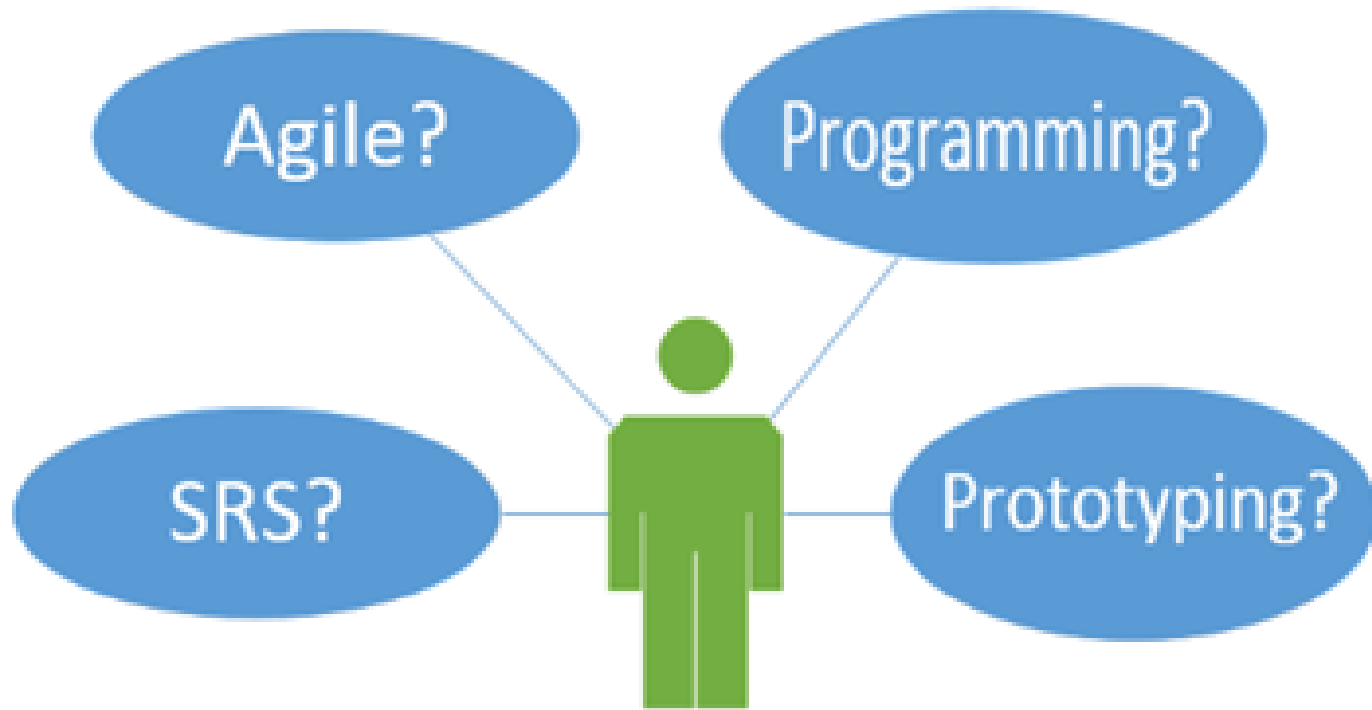




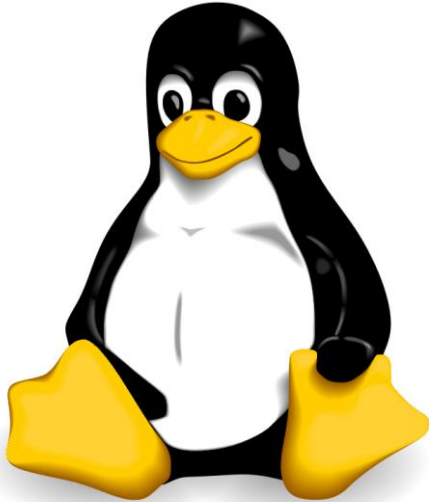
ИСПОЛЬЗОВАНИЕ HTML- ПРОТОТИПОВ ДЛЯ РЕВЕРСИВНОГО АНАЛИЗА ТРЕБОВАНИЙ: ЗА И ПРОТИВ

Николай Киреев
Студия WebMax.BY
www.webmax.by

Что выбрать?



Линус Торвальдс о функциональных спецификациях:



«... Они практически бесполезны. Я не видел ни одного крупного проекта, в котором спецификации полностью соответствовали реальности и при этом действительно помогли разработчикам.

Зато я видел множество неудавшихся проектов, которые слепо полагались на спецификации.

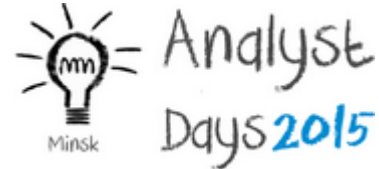
Поверьте мне: разработка по спецификации — худший способ создания приложений, так как он подразумевает, что ваш проект будет делаться только по теории, без оглядки на реальность.» [1]

6 причин по которым не стоит писать функциональные спецификации:

Эссе из книги «[Getting Real](#)», написанной в компании 37signals. [2]

- 1. Спецификация — это фикция.** Она не имеет никакого отношения к реальности... Спецификация никак не приближает проект к завершению — ведь это не более чем слова на бумаге.
- 2. Задача спецификации — угодить всем.** В спецификации никогда не идет речь о нахождении сложных компромиссов и выборе оптимальных вариантов, а при разработке приложений без этого не обойтись.
- 3. Спецификация — лишь иллюзия соглашения.** Множество заверяющих подписей под несколькими страницами спецификации — это не соглашение.

6 причин по которым не стоит писать функциональные спецификации:



Эссе из книги «[Getting Real](#)», написанной в компании 37signals.

4. Спецификация заставляет вас принимать самые важные решения тогда, когда вы меньше всего знаете о проекте.

Меньше всего о проекте известно в самом начале работы. Так почему вы должны принимать самые важные решения, даже не приступив к работе?

5. **Использование спецификаций приводит к обрастанию ненужной функциональностью.** На момент написания спецификации вас ничего не ограничивает, кроме вашей фантазии. Нет ничего легче, чем дописать абзац текста или добавить еще один пункт в список.

6. **Спецификация не позволяет вам развиваться, меняться и оценивать пройденный путь.** Само существование фиксированной спецификации противоречит тому факту, что требования могут меняться в процессе создания приложения.

Функциональные спецификации «без фанатизма»:

ЯВЛЯЮТСЯ:

- основой для приблизительной оценки трудоёмкости и стоимости проекта;
- промежуточным результатом в процессе разработки ПО;
- **архитектуро-определяющими и служат источником для проектирования архитектуры ПО;**
- **источником для внесения изменений в архитектуру ПО в процессе эксплуатации.**

и НЕ являются:

- продуктом, в котором заинтересованы заказчики и пользователи;
- основной целью аналитиков и проектировщиков;
- основной целью процесса разработки;
- полной и достаточной информацией о программном продукте;
- стабильными и очевидными как в процессе разработки, так и в дальнейшей эксплуатации.

Business Requirements

Цели инвесторов
(Business Requirements)

скрытые

Уровень «облачности»

очевидные

Уровни «прямой
видимости»

Пользовательские
требования
(User Requirements)

очевидные

скрытые

Уровень «тумана»

Software Functional Requirements

Функции бизнес-логики,
инициируемые пользователями через GUI

Функции GUI, обеспечивающие UX

Функции бизнес-логики,
инициируемые через внешние интерфейсы (порты)

Функции типовой архитектуры

«Подводные» уровни

Front-end-
уровни

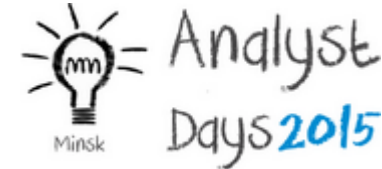
Back-end-уровни

Уровни видимости
функциональных
требований [4]

Manifesto

Become a Programmer, Motherfucker

Buy a T-Shirt, Asshole!



Programming, Motherfucker

Do you speak it?

We are a community of motherfucking programmers who have been humiliated by software development methodologies for years.

We are tired of *XP, Scrum, Kanban, Waterfall, Software Craftsmanship* (aka *XP-Lite*) and anything else getting in the way of...*Programming, Motherfucker.*

We are tired of being told we're autistic idiots who need to be manipulated to work in a Forced Pair Programming chain gang without any time to be creative because none of the 10 managers on the project can do... *Programming, Motherfucker.*

We must destroy these methodologies that get in the way of...*Programming, Motherfucker.*

* * * *

Our Values

They Claim To Value	They Really Value	We Fucking Do
Individuals and interactions	<i>Tons of billable hours</i>	Programming, Motherfucker
Working software	<i>Tons of pointless tests</i>	Programming, Motherfucker
Customer collaboration	<i>Bleeding clients dry</i>	Programming, Motherfucker
Responding to change	<i>Instability and plausible deniability</i>	Programming, Motherfucker

We think the shit on the left, is really just the con in the middle, and that we really need to just do the thing on the right...*Programming, Motherfucker.*

Signed,

Zed A. Shaw

And The Programming Motherfuckers

**Может лучше
сразу писать
код, чем
«жевать»
требования?**

*Пишем и...
переписываем,
снова, снова и
снова...*

**Манифест «крутых»
программеров от
Зеда Шоу [3]**



Разработка динамических прототипов вместо функциональных спецификаций и кодирования

Процесс разработки требований прототипированием

1. ИНИЦИАЦИЯ

- Определяем пользователей (роли, включая абстрактные) и основные функциональные сервисы (Use Cases), которые должна им предоставить система.
- Расставляем приоритеты и составляем список последовательности разработки итераций
- Согласуем краткие описания, по необходимости наброски интерфейсов (wireframe) для уникальных (не имеющих близких аналогов) сервисов.

Процесс разработки требований прототипированием

2. РЕАЛИЗАЦИЯ ПРОТОТИПА (ИТЕРАЦИЯ)

- Реализуем действующий html-прототип очередного подлежащего разработке сервиса, добиваясь (по возможности) полной имитации его функционирования.
- Тестируя, отрабатываем основной и альтернативные потоки, а также меры по исключению ошибок пользователей.
- Каждый разрабатываемый html-прототип размещается в облаке и представляется для тестирования заказчику
- Внесение изменений (по результатам тестирования) в прототип и окончательное согласование функциональности. Завершение цикла.

Процесс разработки требований прототипированием

3. ЗАВЕРШЕНИЕ ПРОТОТИПИРОВАНИЯ

- По завершению последней итерации, по необходимости, создаются детальные UML-модели, пишутся сценарии, спецификации требований
- Отдельные сервисы интегрируются в общую оболочку
- Последующая разработка приложения производится в рамках стандартных моделей



Области применения методики

1. Для приложений с высоконагруженными графическими интерфейсами
2. Для инновационных разработок, не имеющих близких аналогов
3. Для приложений в которых следует исключать риски ошибочных действий пользователей
4. При неясных, нестабильных и изменяющихся требованиях

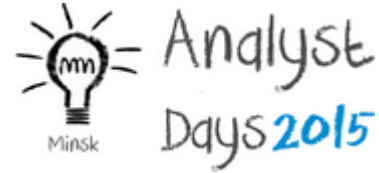
ЗА & ПРОТИВ

ПРЕИМУЩЕСТВА ПРОТОТИПИРОВАНИЯ	НЕДОСТАТКИ vs Спецификация + Wireframes	НЕДОСТАТКИ vs AGILE
1. Функционал определятся не «на бумаге», а тестированием действующего прототипа	1. Требуется затрат времени	1. Требуется затрат времени на прототип, а не на приложение
2. Определяются функции бизнес-логики и UX, уменьшающие ошибки пользователей	2. Требуется высокой квалификации	2. Код прототипа не используется в последующей разработке
3. Лёгкость внесения изменений	3. Относительно узкая область применения	3. Усложняется процесс управления проектом
4. Интегрируется в различных моделях разработки		
5. Возможность распараллеливания работ		

ПРИМЕР ПРОТОТИПИРОВАНИЯ

<http://www.webmax.by/docs/shicht/home.html>

Список используемой литературы



1. Линус Торвальдс

http://gettingreal.37signals.com/ch11_Theres_Nothing_Functional_about_a_Functional_Spec.php

2. Эссе из книги «Getting Real» (<http://gettingreal.37signals.com/>),
перевод ресурс «Хорошие IT-статьи» <http://factorized.tumblr.com/>

3. Зед А. Шоу (Zed A. Shaw) Manifesto <http://programming-motherfucker.com/>

4. Н. Киреев Что делать, когда даже Agile «не рулит?»
<http://analyst.by/articles/что-делат-когда-даже-agile-не-рулит>

Спасибо за внимание!

Николай Киреев

студия WebMax.BY

info@webmax.by

www.webmax.by

*Вебинары-тренинги аналитиков
WebMax.BY*