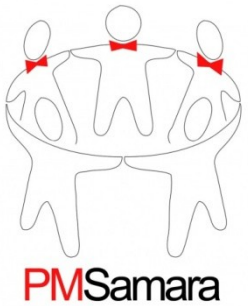
Независимая
научно-практическая конференция
«Разработка ПО 2011»

31 октября - 3 ноября, Москва

Навыки менеджера небольшого проекта. Окопная правда Или экзамен по проектовождению



Калугин Александр

Здрасьте, это я!

- Click to edit Master text styles
  - Second level
    - Third level
      - Fourth level
        - Fifth level

Ph.D, PMP

MERCURY DEVELOPMENT

Менеджер менеджеров

PMARCOR

Автор
http://pmarcor.com/

PMSamara

Соорганизатор
http://pmsamara.blogspot.com/

Disclaimer

- Доклад - точка зрения.
- Обобщение моей практики и моих коллег.
- Может не сработать.
- Речь пойдет именно о ежедневной практике и мелкой технике.

Вася

- Хочу пройти подготовку и сдать экзамен на права вождения проектного транспортного средства

- На какую категорию будете сдавать? Какое у вас транспортное средство?

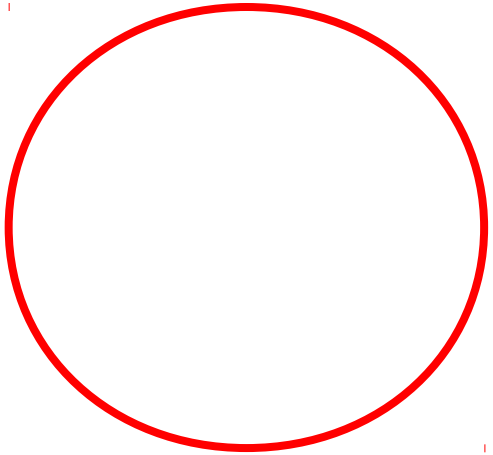
- Вот техпаспорт…

Инспектор

Технический паспорт проекта. Лицевая сторона

- **Тип**: Fixed price/Fixed scope/Fixed time.
- **Марка:** Разработка компактного функционального компонента.
- **Объем двигателя**: *Проект, который помещается в голове.*
- **Ходовая**: Команда до 5-7 человек. Команда придана проекту. Нет противодействия со стороны команды.
- **Срок эксплуатации**: До 3-х месяцев.
- **Условия эксплуатации**: Характеристики внешней среды не изменяются. Нет внешнего противодействия.

Технический паспорт.
Оборотная сторона (голограмма)



Вася

- Ну, что скажете?

- Понятно, у Вас *небольшой проект*

- Вот билеты. Идите готовьтесь.

Инспектор

Билет 1. Трансмиссия

- **ВОПРОС:** Какие коробки передач подойдут для выбранного типа проектного транспортного средства?

- **ВАРИАНТЫ ОТВЕТА:**
  1. Agile
  2. Формальные методологии
  3. Автоматические коробки передач
  4. Специальная коробка…

1.1. Agile

- Нет времени на эволюцию.
  (Фактически - одна итерация)
- Нет смысла в эволюции.
- Agile Manifesto – ***да***!
  - Личности и их взаимодействия важнее,
    чем процессы и инструменты;
  - Работающее программное обеспечение важнее,
    чем полная документация.
- Agile Manifesto – ***нет*** !
  - Сотрудничество с заказчиком важнее, чем контрактные обязательства;
  - Реакция на изменения важнее, чем следование плану.

1.2. *Формальный* процесс

- C’est trop!
  - Не нужен разработчикам;
  - Скорее всего, слишком дорого;
  - Не соответствует бизнес-цели.
- Но! Владение методологией – полезно
  - Для общения со внешней средой:
    - Разговор с заинтересованными лицами с применением терминологии PMBOK,
    - RUP и use-cases в обработке требований, и др.
  - Задают некоторую *матрицу* в работе менеджера.

1.3. Автоматизированные средства и метрики

- ***Не нужны:***
  - Не требуется агрегированное представление информации - можно работать «с первичкой».
  - Визуализацию и удобный интерфейс для принятия управленческих решений – да, но проект умещается в голове.
  - Метрики не работают, так как слишком маленькая выборка.

- ***Но!***
  - Трекер – может пригодится. Может использоваться для всего.
  - Метрики по *переоткрытым дефектам* как эффективное средство диагностики.

1.4. Специальная коробка

- Менеджер-лидер, а не администратор.
- Коммуникация и еще раз коммуникация.
- Команда – это не множество экспертов.
  Формализм и гейты – скорее всего не пройдут.
  Совместная ответственность за результат.
- Ежедневные совещания – это хорошо.
- Основной способ передачи информации – устно.
- Задачи в трекер. Требования в трекер. Дефекты в трекер. Вместо трекера даже табличка в Google Spreadsheet – подойдет.
- Горизонт планирования в деталях – неделя.
- Играющий тренер – это хорошо.
- Baby-Sitting – плохо, но теоретически возможно.

Вася

- ОК. С процессом понятно. А вообще, какие главные приоритеты у менеджера небольшого проекта
  - Преодоление опасностей (успех проекта, требуемый функционал, в рамках бюджета, в срок)
  - Удовлетворенность заказчика/Успех продукта
  - Visibility
- А о каких опасностях идет речь?
  - Об этом в следующем билете

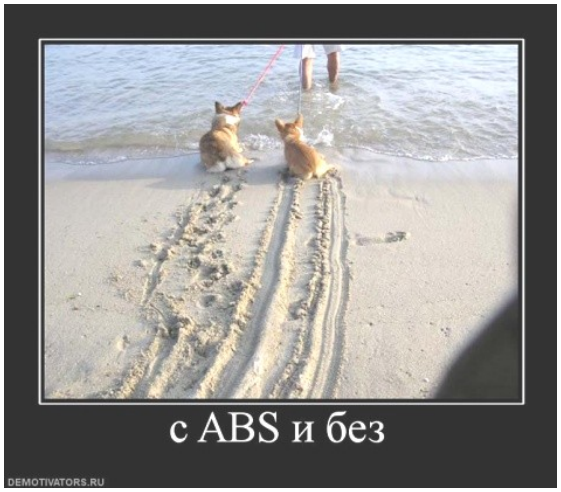
Инспектор

Билет 2. Опасности на дороге

- **ВОПРОС:** Какие основные опасности на дороге для такого проекта?

- **ВАРИАНТЫ ОТВЕТА:**
  1. Нехватка времени
  2. «Недоспек»
  3. Технические риски
  4. Организационные риски
  5. Недопонимание с командой
  6. Внешние риски/политические вопросы
  7. Проблемы коммуникации между участниками

2.1. Нехватка времени



- ***Проблемы:***
  - Разработчиков много – менеджер один;
  - Глубокое планирование – всё таки не хватает; гибкости. Полностью на откуп команде – вряд ли…
  - Соблазн «*покодить».*
- ***Решения:*** *Дать возможность команде восполнять пробелы без менеджера, понимая конечные цели.*
  - Чёткое понимание бизнес-целей. Создание приоритетов и миссии проекта. Kick-off.
  - Оценка менеджером ситуации на проекте, обратная связь команде – регулярно. Положительная и/или отрицательная.
  - Разграничение ответственности между участниками.
  - Ревью состояния кода и продукта – ***объем позволяет!***
  - «Набегающая волна». Для того чтобы выполнить задачу, мелочи не важны
  - Бэклог дополнительных задач менеджера. ***Всё в трекер!***

2.2. «Недоспек»

- ***Примеры:***
  - Неоднозначные требования/ Отсутствие требований.
  - Различные интерпретации тестировщиками и программистами.
- ***Решение:*** *Обнаружить как можно раньше*
  - Замыкание коммуникативных процессов на менеджера.
  - Участие команды тестировщиков в анализе спецификации на ранних этапах.
  - Разработка чеклиста на ранних этапах.
  - Сравнение с аналогами.
  - Утверждение Invalid и Wontfix багов непосредственно менеджером.
  - Обсуждение спецификации только в письменном виде.
  - Регулярные демонстрации заказчику и обсуждения.
  - Обсуждение архитектуры с заказчиком проекта.

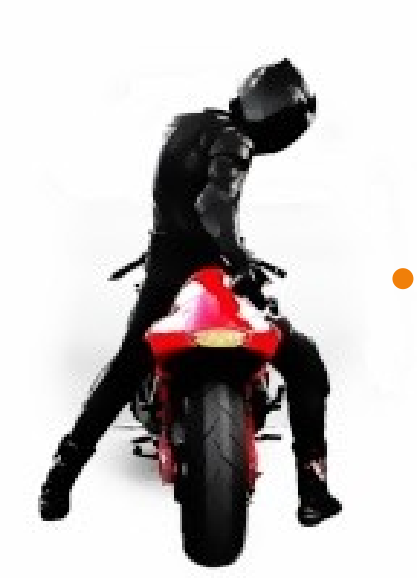
2.3. Технические риски

- ***Примеры:***
  - Компоненты работают не так как описано.
  - Ошибки в архитектуре, «неучет» нефункциональных требований, и др.
- ***Решения:*** *Обнаружить как можно раньше*
  - Начинаем с функциональности, связанной с наибольшим риском или наиболее важной функциональности.
  - В первую очередь реализуем компоненты, соответствующие протоколам общения между модулями и интеграцией с внешними компонентами.
  - Собрать продукт до рабочей версии как можно раньше, чтобы проверить всё в сборе.
  - Говорим о рисках и о технических рисках на ежедневном совещании.
  - Менеджер – кодит сам.

2.4. Организационные риски

- ***Пример:***
  - Болезнь ведущего разработчиков и/или тестировщиков.
- ***Решения:***
  - Быть тех-лидом.
  - Кодить вместе:
    - Разбиение системы на модули;
    - Интерфейсы и протоколы между компонентами;
    - TDD;
    - Атомарность коммитов.
  - Предотвращение *феодального владения кодом*:
    - Постановка задачи двоим (архитектор-ведомый или двое равных);
    - Чередование этапов выполнения одной задачи между несколькими исполнителями.;
    - Варианты парного программирования.

Вася



- А если я бывший разработчик, какие специальные навыки нужны менеджеру небольших проектов?
  - Хмм… водительские…

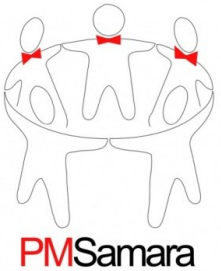
Инспектор

Навыки

- Навыки из «прошлой жизни»
  - Здравый смысл.
  - Чутьё. Понимание основ архитектуры ПО. Понимание, что такое плохие требования, что такое технические риски
  - Умение разрабатывать или тестировать.
  - «Драйв». Лидерские качества. Авторитет у разработчиков.
- Новые навыки
  - Понимание работы других ролей.
  - Time-management.
  - Понимание бизнес-целей и приоритетов.
  - Коммуникация: умение давать конструктивную оценку .происходящему, обратную связь команде.
  - Формальные процессы, коммуникация с заказчиком.

Экзамен

- По результатам экзамена, Василию выданы **права на вождение небольшого проекта.**



Спасибо!

Калугин Александр

info@pmarcor.com

http://pmarcor.com/ http://pmsamara.com

@pmarcor

Ваши вопросы?