Программная инженерия, лекция 2: Унифицированный процесс

Пожидаев М. С.

12 сентября 2022 г.

Требования

Что должен уметь унифицированный процесс?

- Обеспечивать руководство деятельностью команды;
- управлять задачами отдельного разработчика и команды в целом;
- указывать, какие артефакты следует разработать;
- предоставлять критерии для отслеживания и измерения продуктов и функционирования проекта.

Процесс разработки — это сумма различных видов деятельности, необходимых для преобразования требований пользователей в программную систему.

Use cases

UP управляется вариантами использования

Вариант использования — это часть функциональности системы, необходимая для получения пользователем значимого для него, ощутимого и измеримого результата.

Кантианский источник знаний

Варианты использования обеспечивают функциональные требования.

Сложность в формализации вариантов использования заключается в том, что *их нужно перечислить все!*.

Не одно и то же

Use case не функция программы!

Так думали раньше, но сейчас уже нельзя!

Считается, что описание функций может ответить на вопрос: «что система делает?».

Следует вспомнить про роль пользователя

Подход вариантов использования добавляет три слова: «для каждого пользователя».

Варианты использования являются основой для написания тестов.

Типы проектов

Use cases нужны всем

Следует представить возможные типы вариантов использования для проектов таких типов:

1. Настольные приложения.
2. Библиотеки (например, `libssl`).
3. Сетевые сервисы (особенно для кроулеров и `React`).
4. Утилиты командной строки.

Архитектура

UP ориентирован на архитектуру

Понятие архитектуры программы включает в себя наиболее важные статические и динамические аспекты системы. Она зависят от:

- платформы для работы (компьютерной архитектуры);
- операционной системы, СУБД, сетевых протоколов;
- доступности готовых блоков, особенностей развертывания;
- нефункциональных требований (например, производительности и надежности).

Архитектура $\leftrightarrow$ use cases

Как они взаимосвязаны?

Каждый продукт имеет функции и форму. Одно без другого не существует.

Основой для архитектуры является анализ вариантов использованяя.

Если из вариантов использования невозможно выбрать архитектуру, варианты использования построены неверно.

Платформа!

Платформа — часть архитектуры, не связанная с вариантами использования.

Варианты использования и архитектура *равноправны*.

Рождение архитектуры

Как работает архитектор

1. Создает грубый набросок архитектуры, начиная с платформы.
2. Работает с подмножеством выделенных вариантов использования, каждый из которых соответствует одной из ключевых функций системы.
3. Каждый из выбранных вариантов использования детально прорабатывается и конкретизируется в последовательность работ.

Примеры архитектур

Монолитная и микросервисная

1. Монолитная: всё выполняется в одном процессе (и падает тоже сразу всё).
2. Микросервисная: всё разбито на контейнеры.

Итеративно и инкрементно

Что происходит внутри итерации?

Разработчики выбирают задачи, которые должны быть решены в ходе итерации, под воздействием двух факторов:

1. В ходе итерации следует работать с группой вариантов использования, которая повышает применимость продукта в ходе дальнейшей разработки.
2. В ходе итерации следует заниматься наиболее серьезными рисками.

Оценки итераций

И критерии успешности

Итерация должна увеличивать количество полезных работающих функций. Это очевидно.

Но если...

...итерация не выполнила своей задачи, разработчики должны пересмотреть свои решения и попробовать другой подход. ‘Это допускается.

Для получения максимальной экономии команда должна выбирать только те итерации, которые требуются для выполнения цели проекта. Для этого следует расположить итерации в логической последовательности.

жизненный цикл

Цикличность разработки

Унифицированный процесс циклически повторяется. Эта последовательность повторений унифицированного процесса представляет собой жизненный цикл системы.

Результат цикла

Каждый цикл завершается поставкой выпуска продукта заказчикам!

Новый цикл

Требования для начала цикла

Для того чтобы эффективно выполнять новый цикл, разработчикам необходимо знать:

- модель вариантов использования;
- модель анализа для уточнения вариантов использования и создания первичного облика поведения системы;
- модель проектирования (подсистемы, классы и интерфейсы);
- модель реализации;
- модель развертывания.

Фазы

Внутри каждой фазы руководители или разработчики могут потерпеть неудачу в работе, — но только на данной итерации и в связанном с ней приращении.
Каждая фаза заканчивается вехой.
Унифицированный процесс предусматривает четыре фазы:

- анализ и планирование требований;
- проектирование;
- построение (реализация);
- внедрение.

Анализ требований

В чём особая роль этой фазы?

Вход

Хорошая идея.

Выход

Концепция готового продукта и *бизнес-план*.

Спасибо за внимание!

Всё о курсе: https://marigostra.ru/materials/engineering.html
E-mail: msp@luwrain.org
Канал в Телеграм: https://t.me/MarigostraRu