Методологии разработки ПО

Лекция 5
Базовые методологии разработки ПО (ч.2)

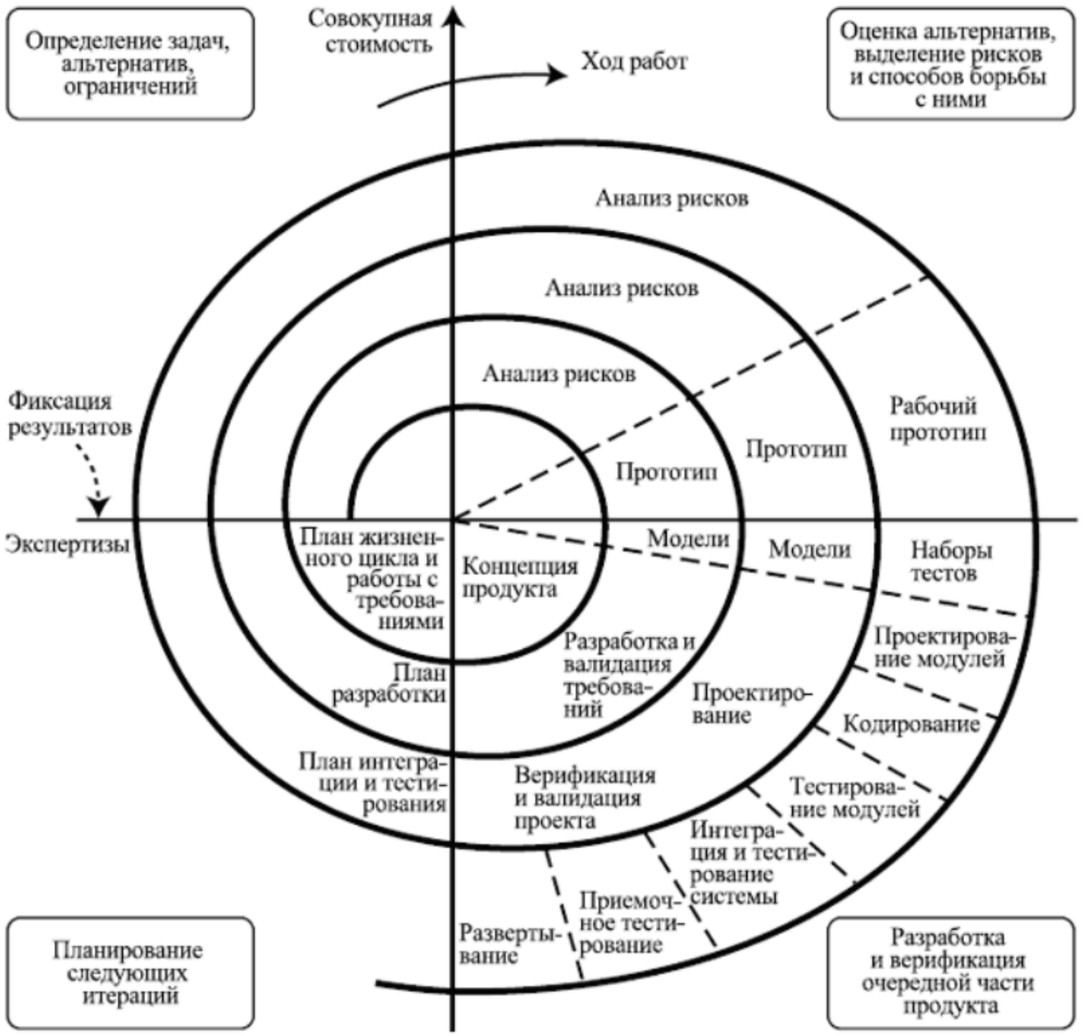
Старичков Н.Ю., ФКН ВШЭ, 2022/2023 уч.год

СПИРАЛЬНАЯ МОДЕЛЬ

СПИРАЛЬНАЯ МОДЕЛЬ

Этапы:

- Планирование
- Анализ рисков
- Конструирование
- Оценка результатов



Плохой пример №1

Допустим, мы делаем какое-то инновационное решение

Плохой пример №1

Например, разрабатываем автоматический генератор песен

Плохой пример №1

Например, разрабатываем автоматический генератор песен

- Кажется, что тут более-менее понятно, что именно мы хотим сделать
- Тут можно как-то постараться оценить сроки
- Тут есть очень большой риск не сделать проект

Плохой пример №1

Например, разрабатываем автоматический генератор песен

- Кажется, что тут более-менее понятно, что именно мы хотим сделать
- Тут можно как-то постараться оценить сроки
- Тут есть очень большой риск не сделать проект
- Либо же мы проект сделаем, но он окажется никому не нужным

Плохой пример №1

Например, разрабатываем автоматический генератор песен

- Кажется, что тут более-менее понятно, что именно мы хотим сделать
- Тут можно как-то постараться оценить сроки
- Тут есть очень большой риск не сделать проект
- Либо же мы проект сделаем, но он окажется никому не нужным

Проблема в том, что тут проект сам по себе - риск

Плохой пример №1

Например, разрабатываем автоматический генератор песен

- Кажется, что тут более-менее понятно, что именно мы хотим сделать
- Тут можно как-то постараться оценить сроки
- Тут есть очень большой риск не сделать проект
- Либо же мы проект сделаем, но он окажется никому не нужным

Проблема в том, что тут проект сам по себе - риск
И даже использование спиральной модели конкретно эти риски никак не снимет

Плохой пример №2

Допустим, мы делаем софт для автоматизации вуза…

- Понимаем, что нужно сделать
- Понимаем, когда это нужно сделать

Плохой пример №2

Допустим, мы делаем софт для автоматизации вуза…

- Понимаем, что нужно сделать
- Понимаем, когда это нужно сделать
- У нас есть подходящая команда?
- Сроки адекватные?

Плохой пример №2

Допустим, мы делаем софт для автоматизации вуза…

- Понимаем, что нужно сделать
- Понимаем, когда это нужно сделать
- У нас есть подходящая команда?
- Сроки адекватные?

Если на оба вопроса выше ответ «да» - то использование спиральной модели - это overkill

Плохой пример №3

Допустим, что мы делаем свой фоторедактор

- Спиральная модель предполагает уточнение целей и планов на каждой итерации
- В том числе, и с учетом мнений пользователей

Плохой пример №3

Допустим, что мы делаем свой фоторедактор

- Спиральная модель предполагает уточнение целей и планов на каждой итерации
- В том числе, и с учетом мнений пользователей

Проблема тут в том, что может получится так, что из-за лишних итераций мы очень сильно «удлиним» проект и сделаем много лишней работы

Плохой пример №3

Допустим, что мы делаем свой фоторедактор

- Спиральная модель предполагает уточнение целей и планов на каждой итерации
- В том числе, и с учетом мнений пользователей

Проблема тут в том, что может получится так, что из-за лишних итераций мы очень сильно «удлиним» проект и сделаем много лишней работы

Тут можно было бы вполне обойтись одной «итерацией»

Плохой пример №3

Допустим, что мы делаем свой фоторедактор

- Спиральная модель предполагает уточнение целей и планов на каждой итерации
- В том числе, и с учетом мнений пользователей

Проблема тут в том, что может получится так, что из-за лишних итераций мы очень сильно «удлиним» проект и сделаем много лишней работы

Т.е. риск в том, что мы получим бесконечный «долгострой»

Хороший пример №1

Допустим, мы делаем высокопроизводительную систему для обработки пользовательских запросов

- Мы понимаем, что именно нужно сделать
- Мы понимаем, как в целом мы будем это делать

Хороший пример №1

Допустим, мы делаем высокопроизводительную систему для обработки пользовательских запросов

- Мы понимаем, что именно нужно сделать
- Мы понимаем, как в целом мы будем это делать
- Но в процессе работы мы можем столкнуться с тем, что у нас:

Хороший пример №1

Допустим, мы делаем высокопроизводительную систему для обработки пользовательских запросов

- Мы понимаем, что именно нужно сделать
- Мы понимаем, как в целом мы будем это делать
- Но в процессе работы мы можем столкнуться с тем, что у нас:
  - Команда не в состоянии решить возникающие сложности

Хороший пример №1

Допустим, мы делаем высокопроизводительную систему для обработки пользовательских запросов

- Мы понимаем, что именно нужно сделать
- Мы понимаем, как в целом мы будем это делать
- Но в процессе работы мы можем столкнуться с тем, что у нас:
  - Команда не в состоянии решить возникающие сложности
  - Возникшие сложности требуют увеличения сроков проекта и/или бюджета

Хороший пример №1

Допустим, мы делаем высокопроизводительную систему для обработки пользовательских запросов

- Мы понимаем, что именно нужно сделать
- Мы понимаем, как в целом мы будем это делать
- Но в процессе работы мы можем столкнуться с тем, что у нас:
  - Команда не в состоянии решить возникающие сложности
  - Возникшие сложности требуют увеличения сроков проекта и/или бюджета

  В таком случае спиральная модель подходит отлично - т.к. мы сможем на каждом этапе выяснить, в чем проблемы и постараться их оперативно решить

Хороший пример №2

Допустим, мы делаем софт для Falcon-X

- В таком случае, мы не то что не понимаем, справиться ли наша команда, мы даже не понимаем, с чем мы столкнемся в будущем - с какими сложностями

Хороший пример №2

Допустим, мы делаем софт для Falcon-X

- В таком случае, мы не то что не понимаем, справиться ли наша команда, мы даже не понимаем, с чем мы столкнемся в будущем - с какими сложностями
- После создания очередного прототипа мы будем его проверять и тестировать

Хороший пример №2

Допустим, мы делаем софт для Falcon-X

- В таком случае, мы не то что не понимаем, справиться ли наша команда, мы даже не понимаем, с чем мы столкнемся в будущем - с какими сложностями
- После создания очередного прототипа мы будем его проверять и тестировать
- По итогам итерации тестов и проверок - мы будем понимать в чем сложности

Хороший пример №2

Допустим, мы делаем софт для Falcon-X

- В таком случае, мы не то что не понимаем, справиться ли наша команда, мы даже не понимаем, с чем мы столкнемся в будущем - с какими сложностями
- После создания очередного прототипа мы будем его проверять и тестировать
- По итогам итерации тестов и проверок - мы будем понимать в чем сложности
- Соответственно, сможем понять цели на следующую итерацию и пути достижения этих целей

Хороший пример №2

Допустим, мы делаем софт для Falcon-X

- В таком случае, мы не то что не понимаем, справиться ли наша команда, мы даже не понимаем, с чем мы столкнемся в будущем - с какими сложностями
- После создания очередного прототипа мы будем его проверять и тестировать
- По итогам итерации тестов и проверок - мы будем понимать в чем сложности
- Соответственно, сможем понять цели на следующую итерацию и пути достижения этих целей

То есть тут спиральная модель ложится просто идеально и отлично наши проблемы решает

Хороший пример №3 - самый неочевидный

Допустим, мы делаем проект с командой студентов - тот же мессенджер

- Вроде бы мы отлично понимаем, что именно хотим сделать
- И понимаем, как это сделать

Хороший пример №3 - самый неочевидный

Допустим, мы делаем проект с командой студентов - тот же мессенджер

- Вроде бы мы отлично понимаем, что именно хотим сделать
- И понимаем, как это сделать
- И даже компетенции команды не вызывают сомнений и не несут рисков

Хороший пример №3 - самый неочевидный

Допустим, мы делаем проект с командой студентов - тот же мессенджер

- Вроде бы мы отлично понимаем, что именно хотим сделать
- И понимаем, как это сделать
- И даже компетенции команды не вызывают сомнений и не несут рисков

В чем тогда тут проблема? Зачем нужна спиральная модель?

Хороший пример №3 - самый неочевидный

Допустим, мы делаем проект с командой студентов - тот же мессенджер

- Вроде бы мы отлично понимаем, что именно хотим сделать
- И понимаем, как это сделать
- И даже компетенции команды не вызывают сомнений и не несут рисков

В чем тогда тут проблема? Зачем нужна спиральная модель?

А проблема в самих студентах и организации их работы

Хороший пример №3 - самый неочевидный

Допустим, мы делаем проект с командой студентов - тот же мессенджер

- Вроде бы мы отлично понимаем, что именно хотим сделать
- И понимаем, как это сделать
- И даже компетенции команды не вызывают сомнений и не несут рисков

В чем тогда тут проблема? Зачем нужна спиральная модель?

А проблема в самих студентах и организации их работы

В данном случае спиральная модель позволит нам относительно короткими итерациями получать промежуточные версии

Хороший пример №3 - самый неочевидный

Допустим, мы делаем проект с командой студентов - тот же мессенджер

- Вроде бы мы отлично понимаем, что именно хотим сделать
- И понимаем, как это сделать
- И даже компетенции команды не вызывают сомнений и не несут рисков

В чем тогда тут проблема? Зачем нужна спиральная модель?

А проблема в самих студентах и организации их работы

В данном случае спиральная модель позволит нам относительно короткими итерациями получать промежуточные версии
И на каждую следующую итерацию собирать обновленную команду и/или по-другому распределять задачи между ними

RAD-МОДЕЛЬ

RAD-МОДЕЛЬ

Rapid Application Development Model

- **Различные модули разрабатываются различными командами**
- **Жестко ограниченное время**
- **Интеграция отдельных модулей в один проект**
- **Использование инструментов автоматической сборки и генерации кода**

Этапы:

- Бизнес-моделирование
- Анализ и создание модели данных
- Анализ и создание моделей процессов
- Автоматическая сборка приложения и тестирование

>>: tbc…