АО «АЛМАТИНСКИЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»
https://atu.edu.kz/

Факультет «Информационные технологии»
Кафедра «Компьютерная инженерия»

Название курса: «Инструментальные средства разработки программ»



Зиятбекова Гулзат Зиятбеккызы – PhD, ассоциированный профессор
g.ziyatbekova@atu.edu.kz

ОСНОВЫ ПРОЕКТИРОВАНИЯ

ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННЫХ СИСТЕМ

От задачи и границ — к проверяемому решению

УЧЕБНЫЙ МАРШРУТ

Информационная система начинается с задачи

Сначала нужно понять, какую проблему решает система и для кого. Только после этого бизнес-проблема превращается в требования и критерии успеха.

01

Проблема

Что не работает или что должно измениться?

02

Требование

Что система должна делать, чтобы помочь?

03

Успех

Как мы проверим, что проблема решена?

02 · УЧАСТНИКИ

Проектирование связывает людей, процессы и технологии

Каждая роль видит систему со своей стороны. Проектирование переводит эти взгляды в общий язык решения.

ПОТРЕБНОСТЬ ДОЛЖНА ПРОЙТИ ВЕСЬ ПУТЬ ДО РАБОТАЮЩЕЙ СИСТЕМЫ.



Заказчик и пользователи

Формулируют потребность и контекст работы.



Аналитик

Переводит потребность в требования.



Архитектор

Выбирает структуру и границы компонентов.



Команда

Проверяет, реализует и сопровождает систему.

03 · ГРАНИЦЫ

Сначала фиксируют границы системы

Граница отвечает на простой вопрос: что входит в решение, а что остаётся снаружи.

ВНУТРИ ГРАНИЦЫ

Система

- Основная цель проекта
- Пользователи и их действия
- Данные и правила системы

ЗА ГРАНИЦЕЙ

Внешняя среда

- Внешние системы и сервисы
- Ограничения проекта
- Сознательно исключённые функции

ЦЕЛЬ · ПОЛЬЗОВАТЕЛИ · ВНЕШНИЕ СИСТЕМЫ · ОГРАНИЧЕНИЯ

Требования превращают ожидания в проверяемые условия

Фраза «система должна быть удобной» слишком расплывчата. Требование должно быть понято одинаково и проверено на практике.

ХОРОШЕЕ ТРЕБОВАНИЕ УМЕНЬШАЕТ ПРОСТРАНСТВО ДЛЯ РАЗНЫХ ТРАКТОВОК.

ЧТО СИСТЕМА ДЕЛАЕТ

Функциональные требования

Описывают действия и результаты: создать заказ, найти запись, отправить уведомление.

КАК СИСТЕМА РАБОТАЕТ

Нефункциональные требования

Задают скорость, надёжность, безопасность и удобство использования.

ПРОВЕРКА

Критерий качества

Требование можно понять одинаково и проверить по наблюдаемому результату.

Модель показывает систему до написания кода

Модель — это способ вынести важные решения наружу. Разные модели отвечают на разные вопросы и уменьшают неопределённость.



Модель процессов

Какие шаги выполняются и в каком порядке?



Модель данных

Какие объекты существуют и как они связаны?



Модель взаимодействий

Кто и как взаимодействует с системой?

СНАЧАЛА ПРОВЕРЯЕМ ИДЕЮ НА СХЕМЕ, ПОТОМ ЗАКРЕПЛЯЕМ ЕЁ В КОДЕ.

06 · АРХИТЕКТУРА

Архитектура распределяет ответственность

Архитектура задаёт, из каких компонентов состоит система, за что отвечает каждый компонент и как они взаимодействуют.

01

Компоненты

Выделить основные компоненты

02

Роли и интерфейсы

Определить их роли и интерфейсы

03

Слои

Разнести данные и правила по подходящим слоям

04

Проверка рисков

Проверить связи, риски и точки отказа

ХОРОШАЯ СТРУКТУРА ДЕЛАЕТ ИЗМЕНЕНИЯ ЛОКАЛЬНЫМИ, А СВЯЗИ — ПОНЯТНЫМИ.

07 · ДАННЫЕ

ДАННЫЕ — ОБЩИЙ ЯЗЫК СИСТЕМЫ



ПРЕДМЕТНАЯ ОБЛАСТЬ ДОЛЖНА БЫТЬ ПОНЯТНА И ЛЮДЯМ, И ПРОГРАММЕ.

Проектное решение проверяют итерациями

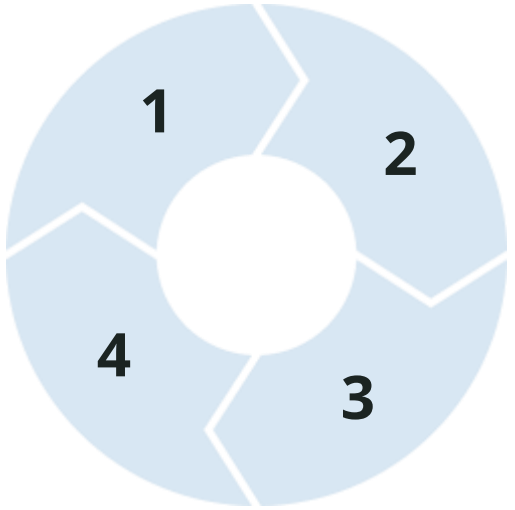
Не нужно ждать окончания реализации, чтобы обнаружить неверную гипотезу. Чем раньше команда получает обратную связь, тем дешевле изменить решение.

Сформулировать

Гипотезу решения.

Собрать

Модель или прототип.



Уточнить

Требования и решение.

Проверить

С пользователями и командой.

ГИПОТЕЗА → МОДЕЛЬ → ПРОВЕРКА → УТОЧНЕНИЕ

09 · ПРЕДУПРЕЖДЕНИЕ

Главные ошибки рождаются до реализации

Код может быть аккуратным, но бесполезным, если неверно понята задача или не проверены ключевые решения.



Размытая цель

Непонятно, какой результат считать успехом.



Требования без владельца

Нет приоритета и ответственного за решение.



Архитектура без запаса

Не учтены нагрузка и будущие изменения.



Проверка в конце

Ошибка обнаруживается слишком дорого.

КАЧЕСТВО РЕАЛИЗАЦИИ НАЧИНАЕТСЯ С КАЧЕСТВА ПРОЕКТИРОВАНИЯ.