Лекция 2

// конец прошлой лекции

Сквозной структурный контроль – набор технических приемов, которые переводят функцию контроля с руководителя на программистов.

Приемы:

- посадить всех программистов вместе (решаются задачи взаимного контроля коллег, взаимного обучения)
- руководитель тоже располагается среди своих подчиненных
- выращивать аналитиков и проектировщиков в самом коллективе
- оцениваем программиста по результатам, наказание финансовое
- организовываем контрольные сессии с докладами о работе (руководитель не приглашается), задача – по максимуму выделить недочеты
- вся подготовленная к сессии раздатка (плакаты и тд) остается в работе в общем зале

Плюсы структурного программирования:

- логические ошибки исправляются на ранних стадиях разработки (на верхних уровнях)
- объединение кодирования, проектирования, тестирования
- комплексная отладка (не нужны отладочные программы для каждой функции каждого модуля)
- взаимодействие с заказчиком с ранних стадий разработки
- много естественных контрольных точек – удобно контролировать выполнение проекта
- удобно локализовать ошибки – код прозрачный, читаемый
- удобно распределять задания между программистами
- отказ в проекте практически сводится к 0
- выделение простых функций на низких уровнях – возможность создавать библиотеки (повторное использование кода)
- практически нет кода в корзине
- плавное распределение ресурсов
- использование базовых логических структур

Проблема – когда программа уходит в разработку оказывается, что что-то заказчику все-таки не нравится и выясняется, что программу нужно модифицировать. Причем разработчики уже будут работать на другом проекте, поэтому дорабатывать будут новые люди.

ПРАВИЛО. Любое изменение написанного кода – снижение его надежности.

// пускай будет интрига

Добавление нового функционала обычно затрагивает данные (нужно менять структуры, добавлять/изменять поля и тд). Воняет кучей проблем.

Со временем программа начинает сыпаться и все пишется заново.

1966 год Хоар “Совместное использование записей”
Три идеи:

- Раз у нас возникает проблема с изменением данного, то давайте теперь изначально выделим, что мы можем делать с этим данным (причем эти действия с другими данными напрямую не работают). При этом данные могут меняться, а набор действий не меняется. **Инкапсуляция** (объединение данных и действий над ними).
- Для внесения изменений делаем надстройку над тем, что есть. **Наследование** + **полиморфизм** (безразличие). Важно, чтобы итоговое объединение выполняло свою функцию в коде.
- Перенос в программу того, как объекты взаимодействуют в физическом мире. Было выделено два вида взаимодействия объектов:
    - Синхронное или акцессорное взаимодействие
    - Событийное или асинхронное взаимодействие

Основные понятия ооп:
Объект – конкретная реализация какого-то абстрактного понятия, обладающая характеристиками состояния, поведения и индивидуальности.
Состояние – одно из возможных вариантов существования объекта.
Поведение – описание объекта в терминах изменения его состояния.
Индивидуальность – сущность, присущая каждому объекту, отличающая его от других.

Модель состояния (модель Мурра) – состоит из множества состояний, множества событий (которые приводят к изменению состояний), правил перехода и действий (с каждым состоянием связано действие, которое переводит объект в это состояние). Такая модель должна быть построена асинхронно.

Категории объектов:

- Реальные объекты – абстракции реальных предметов физического мира
- Роли – абстрактные цели или назначения объекта (один и тот же физический объект может выполнять несколько ролей, но в программе такого быть не должно)
- Инцидент – абстракция чего-либо произошедшего
- Объекты взаимодействия – объекты, получаемые в результате взаимоотношений других объектов (перекресток)
- Объекты спецификации – представление правил, стандартов, критериев качества (расписание движения поездов)

Виды отношений между объектами:

- Отношения старшинства или использования (любой объект выступает в одной из трех ролей – воздействие

    - активный объект – воздействует на других, но на него никто не воздействует
    - посредники – могут как активные, могут как пассивные
    - пассивные – только подвергаются воздействию)
- Отношения включения – один объект может включать в себя другие объекты.