Разработка программных модулей

Тема 1. Жизненный цикл ПО

Тема 2. Структурное программирование

Модели жизненного цикла разработки ПО

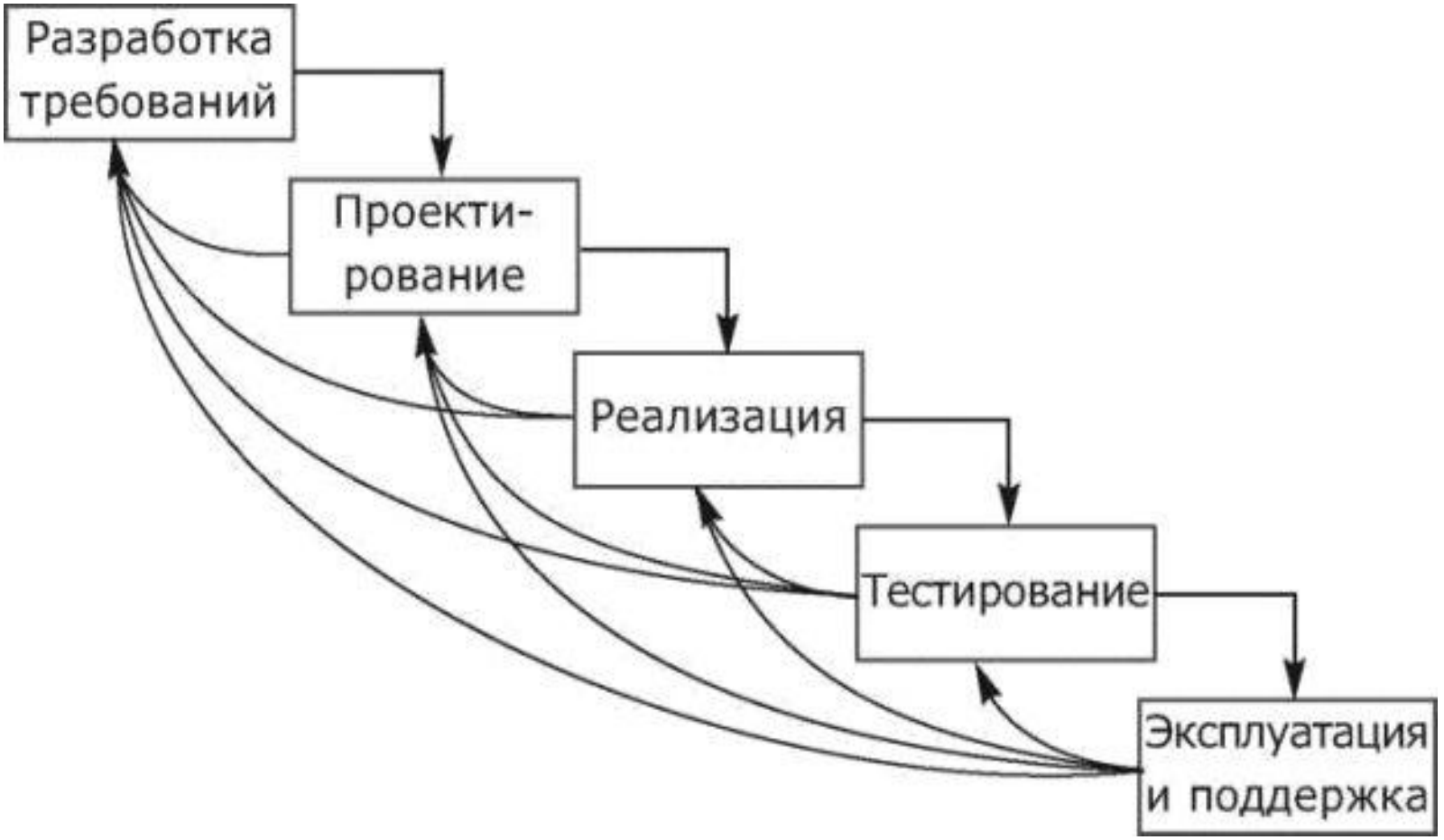
- обобщенная структура, содержащая процессы, действия и задачи, которые осуществляются в ходе разработки, функционирования и сопровождения типового программного продукта в течение всей жизни системы, т.е. от определения требований до завершения ее использования.

Модели ЖЦ ПО

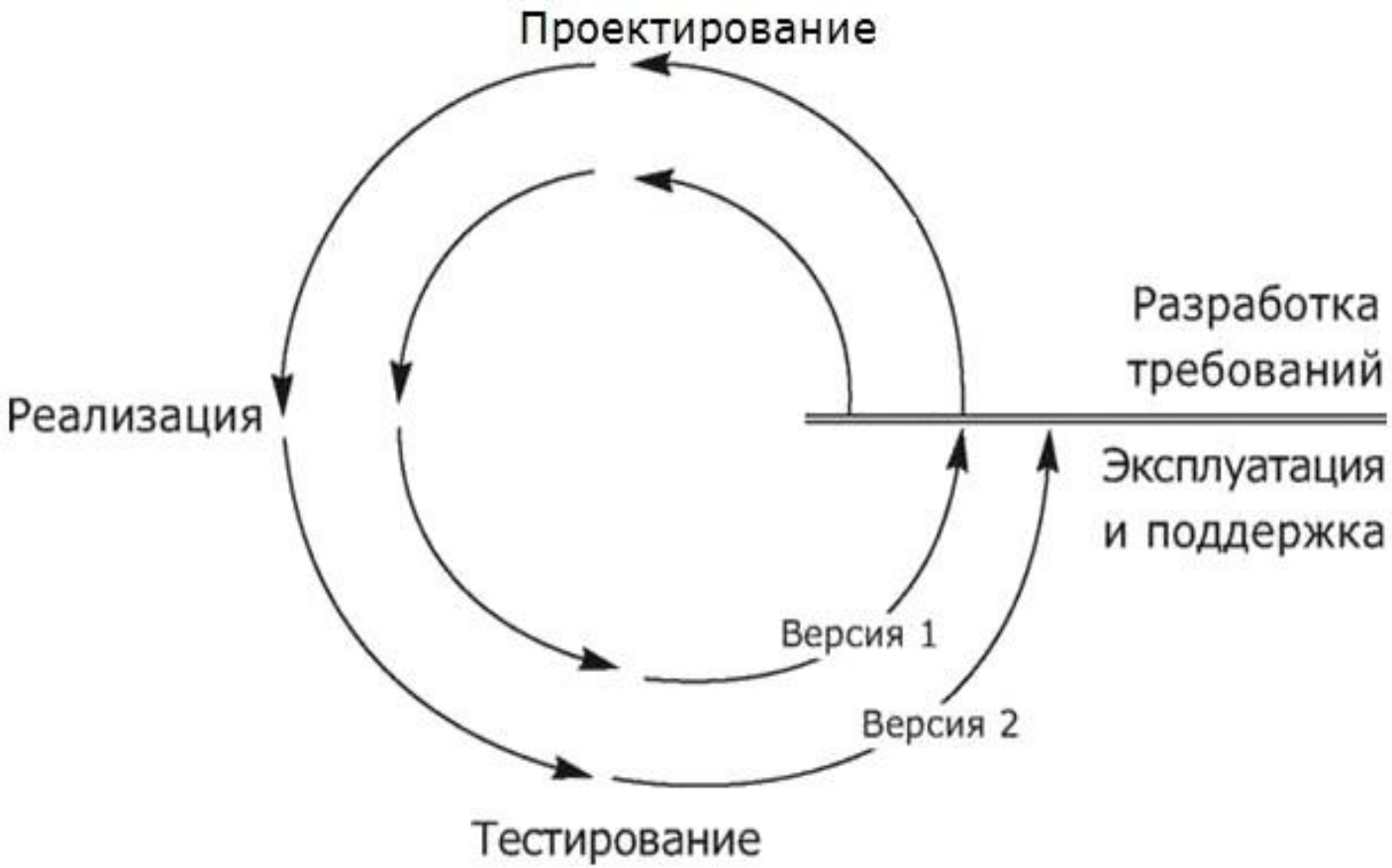
- каскадная модель;
- процессная или поэтапная модель с промежуточным контролем;
- спиральная модель.

Каскадная модель

Поэтапная модель с возвратами



Спиральная модель



Технология структурного программирования

Цели структурного программирования:

- избавиться от плохой структуры программы;
- создавать программы, которые можно было бы понимать, сопровождать и модифицировать без участия автора.

Технология структурного программирования состоит из двух частей:

- нисходящая разработка;
- структурное программирование.

Нисходящую разработку можно рассматривать как процесс, состоящий из трех этапов:

- проектирование;
- планирование;
- реализация.

Цели нисходящей разработки:

- уменьшить сложность программ;
- уменьшить время разработки;
- позволить как можно раньше обнаружить ошибки.

Проектирование программы

- Разбиение программ на модули.

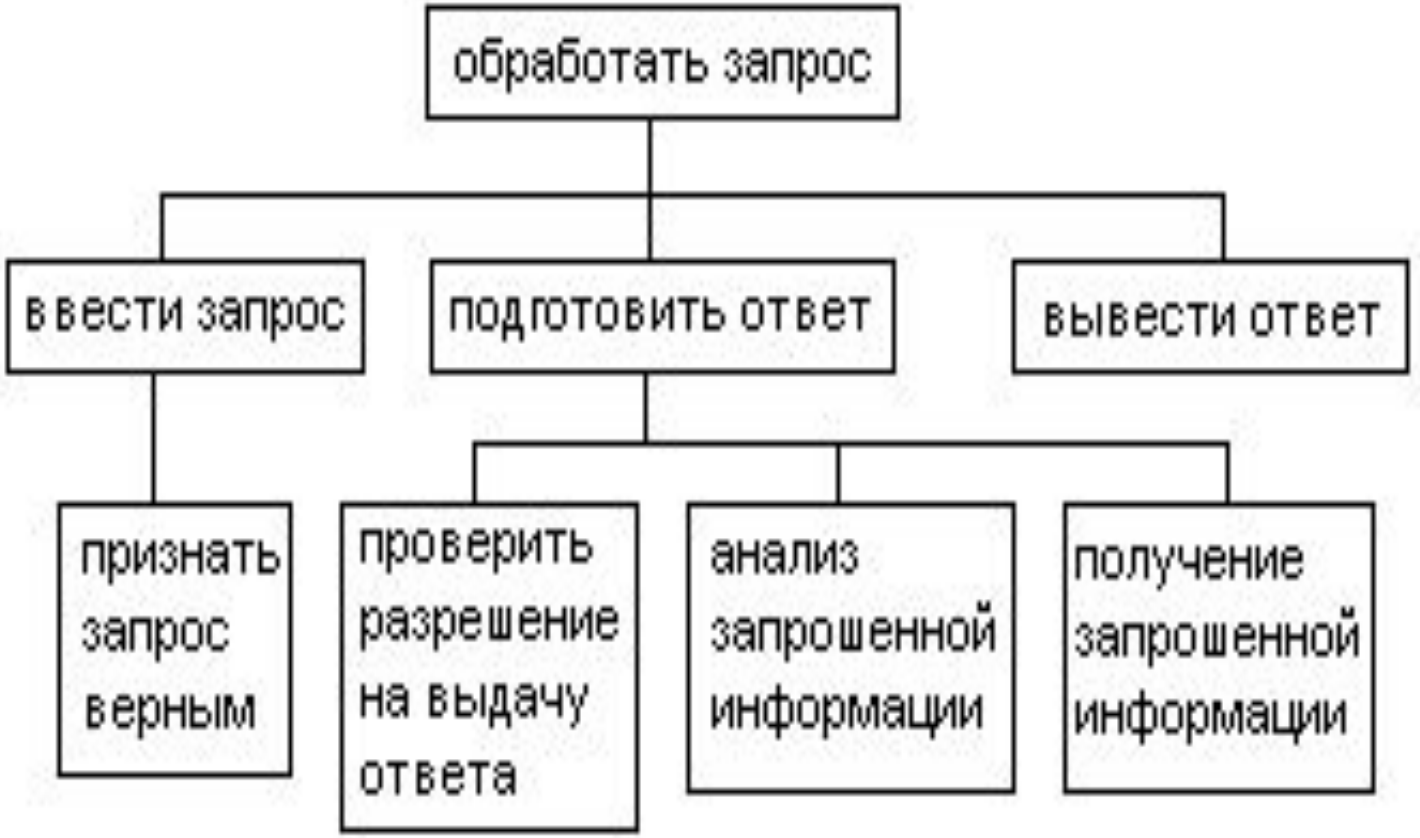
Желательные свойства модулей:

- модуль должен иметь один вход и один выход и возвращать управление тому, кто его вызвал;
- размер модуля должен быть небольшим (10-100 операндов), при этом его легче читать и тестировать;
- модуль не должен сохранять историю своих вызовов для управления своим функционированием;
- модуль должен по возможности реализовывать одну функцию преобразования исходных данных в результат, это позволяет выделять часто употребляемые модули и объединять их в библиотеки;
- по возможности принятие решений в модулях нужно организовать так, чтобы эти решения прямо влияли только на выполнение вызываемых модулей;
- модуль должен быть по возможности независимым от других модулей, для этого важно ослабить связи между модулями.

Проектирование программы

- Абстрагирование – это процесс обобщения, при котором внимание концентрируется на сходстве явлений и предметов и они объединяются в группы на основе этого сходства.
- Уровни абстракции определяют уровни модулей.

Разработка схемы иерархии



Конец этапа проектирования

- известно необходимое число модулей и их спецификации;
- связи между модулями отображены схемой иерархии;
- вся работа проверена пользователями на точность и полноту.

Планирование

На этапе планирования решаются две задачи:

- планирование порядка разработки модуля;
- планирование тестов.

Иерархический подход.

- При этом подходе порядок программирования и тестирования модулей определяется их расположением в схеме иерархии. Сначала программируются и тестируются все модули одного уровня, после чего происходит переход на уровень ниже. При тестировании вызовы модулей нижних уровней заменяются заглушками.

Операционный подход

- При этом подходе модули разрабатываются в порядке их выполнения при запуске готовой программы. Так как порядок исполнения обычно меняется с изменением входных данных, то здесь также остается свобода для выбора порядка разработки.

Планирование тестов

- Вместе с планированием порядка, в котором следует разрабатывать модули, нужно планировать разработку тестов для всего проекта.

Реализация

Этап реализации включает:

- фактическую подготовку тестов;
- программирование и тестирование модулей.

Структурное программирование

Требования:

- текст программы должен быть композицией трех основных элементов: следование, ветвление и цикл;
- употребление goto следует избегать всюду, где это возможно;
- текст программы нужно структурировать, то есть писать так, чтобы можно было легко выделять блоки программы;
- каждый модуль должен иметь один вход и один выход.

Тестирование программного обеспечения

Существует два похода к тестированию.

- Предполагает составлять тесты только на основе внешних спецификаций.
- Предполагает составлять тесты, изучая только логику программы.

Средства автоматизации разработки программм (CASE-средства)

- инструментарий для системных аналитиков, разработчиков и программистов, позволяющий автоматизировать процесс проектирования и разработки программного обеспечения.
- программное средство, поддерживающее процессы жизненного цикла программного обеспечения, включаяанализ требований к системе, проектирование прикладного ПО и баз данных, генерацию кода, тестирование, документирование, обеспечение качества, управление конфигурацией ПО и управление проектом, а также другие процессы (согласно стандарту ISO/IEC 14102:1995).

Особенности средств автоматизации разработки программ:

- поддерживают единственную методологию;
- ориентируются на определенную технологию;
- предназначаются для команд, работающих над единственным проектом;
- используются для разработки информационных систем;
- разрабатываются одной компанией. Возможность интеграции инструментов других компаний отсутствует.

CASE-средствам присущи основные особенности

- наличие мощных графических средств для описания и документирования системы, обеспечивающих удобный интерфейс с разрабочиком и развивающих его творческие возможности;
- интеграция отдельных компонентов CASE-средств, обеспечивающая управляемость процессом разработки ПО;
- использование специальным образом организованного хранилища проектных метаданных (репозитория).

Интегрированное CASE-средство (комплекс средств, поддерживающих полный ЖЦ ПО) содержит следующие компоненты:

- репозиторий, являющийся основой CASE-средства. Он должен обеспечивать хранение версий проекта и его отдельных компонентов, синхронизацию поступления информации от различных разработчиков при групповой разработке, контроль метаданных на полноту и непротиворечивость;
- графические средства анализа и проектирования, обеспечивающие создание и редактирование комплекса взаимосвязанных диаграмм, образующих модели деятельности организации и системы ПО;
- средства разработки приложений, включая языки 4GL (язык 4 поколения) и генераторы кодов;
- средства управления требованиями;
- средства управления конфигурацией ПО;
- средства документирования;
- средства тестирования;
- средства управления проектом;
- средства реверсного инжиниринга ПО и баз данных.

Алгоритм

- Алгоритм - конечный набор правил, который определяет последовательность операций для решения конкретного множества задач и обладает пятью важными чертами: конечность, определённость, ввод, вывод, эффективность.
- Алгоритм - всякая система вычислений, выполняемых по строго определённым правилам, которая после какого-либо числа шагов заведомо приводит к решению поставленной задачи.
- Алгоритм - строго детерминированная последовательность действий, описывающая процесс преобразования объекта из начального состояния в конечное, записанная с помощью понятных исполнителю команд.
- Алгоритм - последовательность действий, направленных на получение определённого результата за конечное число шагов».
- Алгоритм - последовательность действий, либо приводящая к решению задачи, либо поясняющая почему это решение получить нельзя.
- Алгоритм - это точная, однозначная, конечная последовательность действий, которую должен выполнить пользователь для достижения конкретной цели либо для решения конкретной задачи или группы задач за конечное число шагов.

Свойства алгоритма

- *Дискретность.*
- *Определенность.*
- *Результативность* (конечность).
- *Массовость.*

Алгоритмы

- Не для любой задачи существует алгоритм ее решения. Существуют алгоритмически неразрешимые задачи.
- Если алгоритм решения существует, то он может быть неприменим на практике из-за своей высокой сложности.

Сложность алгоритма

- Это количественная характеристика необходимых алгоритму ресурсов для успешного решения задачи.

К ресурсам относятся:

- Время (временная сложность);
- Объем памяти (емкостная сложность).

Сложность алгоритма

- Оценка роста сложности: $n \to \infty$: $T=O(f(n))$
- Фактическая сложность (время работы в секундах) зависит не только от алгоритма, но и от скорости работы компьютера.
- Порядок роста сложности ограничивает размер решаемых задач.

Классификация задач по классам сложности:

- P-сложные - задачи, которые могут быть решены за время, полиномиально зависящее от объёма исходных данных, с помощью детерминированной вычислительной машины.
- NP-сложные - задачи, которые могут быть решены за полиномиально выраженное время с помощью недетерминированной вычислительной машины, то есть машины, следующее состояние которой не всегда однозначно определяется предыдущими.