Стандарты и технология программирования

д.т.н. **Шалфеева**
Елена Арефьевна
shalf@dvo.ru

План изучения дисциплины

- **Процессы и стандарты создания программного обеспечения (или ПрС)**
- Системная инженерия (начальные этапы) и Анализ требований к ПрОбес \ **ПрС**
- Проектирование (design) ПрОбес \ **ПрС**
- Кодирование и Испытания (Тестирование) ПрОбес \ **ПрС**
- Модели процессов программного обеспечения. Управляющие и поддерживающие процессы

Точки зрения на предмет *технологии программирования*

- (1) широкое использование инструментальных средств или
- (2) набор методик и регламентирующих средств, позволяющих, в частности, на каждом этапе провести экспертизу, архивацию и измерение объема и качества проделанной работы (средства – *стандарты*, *процедуры*: соглашения о последовательности действий одного или нескольких разработчиков, направленных на достижение некоторой цели ТП, *программные средства*), или
- (3) *"наука и искусство"* (в качестве средств – *"методы жизненного цикла").*

Виды программного обеспечения

Пример классификации 15-летней давности

- Системное программное обеспечение;
- Программное обеспечение реального времени;
- Инженерное и научное программное обеспечение;
- Встроенное программное обеспечение;
- Текстовые, графические и другие процессоры и редакторы;
- Офисные пакеты;
- Системы управления базами данных;
- Программное обеспечение искусственного интеллекта

Классификатор программ электронных вычислительных машин и баз данных.

Классификация построена на функциональных и технических характеристиках и характеристика сферы эксплуатации ПО

[Приказ Минкомсвязи РФ от 22.09.2020 N 486]

Разделы

- **Встроенное программное обеспечение**
- **Системное программное обеспечение**
- **Средства обеспечения информационной безопасности**
- **Средства разработки программного обеспечения**
- **Прикладное программное обеспечение**
- **Офисные приложения**
- **Лингвистическое программное обеспечение**
- **Промышленное программное обеспечение**
- **Средства управления процессами организации**
- **Средства обработки и визуализации массивов данных**
- **Средства анализа данных**

Технология программирования
(современный взгляд)

Технология программирования это – *"инженерная дисциплина"* с *"инженерными подходами"*.

В качестве ее **средств** рассматриваются *"стандарты, методологии, технологические программные средства, процессы, методы организации, методы управления и системы обеспечения качества"*.

Основной **принцип**, поддерживающий технологию программирования как **инженерную** дисциплину, это **фокусирование внимания на качестве**. Любое применение технологии программирования, означает взятие организацией-производителем обязательств по выпуску **продукции "высокого" качества.**

Качество программного обеспечения определяется (в стандарте ISO 9126) как вся *совокупность его характеристик, относящихся к возможности удовлетворять* высказанные или подразумеваемые *потребности всех заинтересованных лиц*

Качество продукта

Для многих организаций основным критерием деятельности является достижение **высокого уровня качества производимой продукции** либо предоставляемых услуг.

Традиционно продукт считается ***качественным*** в том случае, ***если полностью соответствует техническим требованиям.***

О «проблемах» определения: ***«качественный**, если **соответствует техническим требованиям»***

- 1. Технические требования ориентированы на те **свойства продукта, которые необходимы** заказчику (или пользователю). Однако организация-разработчик может также иметь свои требования к разрабатываемому программному продукту (например, удобство сопровождения), которые обычно не включаются в технические требования заказчика.
- 2. Трудно создать полную спецификацию программного продукта. Поэтому, хотя созданный программный продукт будет полностью соответствовать спецификации, заказчик все равно может не получить высококачественного продукта.

Достижение необходимого уровня качества продукта зависит от того, как осуществляется ***управление качеством*** в компании-разработчике.

Теоретически ***управление качеством*** основывается на принципе определения **стандартов** и процедурных норм, в соответствии с которыми должно разрабатываться **программное обеспечение**, а также на **проверке** выполнения этих норм всеми разработчиками.

В процессе обеспечения качества могут применяться два вида стандартов.

1. *Стандарты на продукцию.*
2. *Стандарты на процесс создания ПрОбес.*

Стандарты в разработке программной продукции

- 1. Стандарты аккумулируют все лучшее из практической деятельности по созданию ПрОбес. Как правило, практические знания приобретаются путем долгого поиска и ошибок. Привнесение этого опыта в определенный стандарт помогает избежать повторения прошлых ошибок. Стандарты в данном случае **собирают знания и опыт**, имеющие значение для организации-разработчика.
- 2. Стандарты предоставляют необходимую основу для реализации процесса обеспечения качества. Имея в наличии стандарты, обобщающие **лучшие знания и опыт**, для обеспечения качества достаточно **контролировать, чтобы они выполнялись** в процессе создания ПрОбес.
- 3. Стандарты незаменимы, когда работа переходит от одного сотрудника к другому. В этом случае деятельность всех специалистов в организации подчиняется единому нормативу. Следовательно, требуется меньше затрат на изучение сотрудником новой работы.

Стандарты на программную продукцию и на процесс ее создания

1. *Стандарты* ***на продукцию.*** Применимы к уже готовым программным продуктам. Они включают стандарты на сопроводительную документацию, например, структуру документа, описывающего системные требования, а также такие стандарты, как, например, стандарты написания программного кода, стандарты ПИФ.

2. *Стандарты* ***на процесс*** *создания ПО.* Определяют ход самого процесса создания программного продукта, например разработку спецификации, процессы проектирования и аттестации. Кроме того, они могут описывать документацию, создаваемую в ходе выполнения этих процессов.

Представление качества **в стандарте ISO 9126**

Качество определяется в ГОСТ Р ИСО/МЭК 9126-93 как «***совокупность признаков и характеристик продукции, относящихся к их способности удовлетворять установленным или предполагаемым потребностям».***

Стандарт ISO 9126 предлагает использовать для описания качества ПО **многоуровневую** модель.

На верхнем **уровне** выделено **6 основных характеристик качества ПО.** Каждая характеристика описывается при помощи нескольких входящих в нее *атрибутов.*

Функциональность

Функциональность (functionality).
Способность ПО в определенных условиях решать задачи, нужные пользователям.
Определяет, что именно делает ПО, какие задачи оно решает.

- ***Функциональная пригодность (suitability).***
  Способность решать нужный набор задач.
- ***Точность (accuracy).***
  Способность выдавать нужные результаты.
- ***Способность к взаимодействию (interoperability).***
  Способность взаимодействовать с нужным набором других систем.
- ***Соответствие стандартам и правилам (compliance).***
  Соответствие ПО имеющимся индустриальным стандартам, нормативным и законодательным актам, другим регулирующим нормам.
- ***Защищенность (security).***
  Способность предотвращать неавторизированный, т.е. без указания лица, пытающегося его осуществить, и не разрешенный доступ к данным и программам.

Надежность

Надежность (reliability).
Способность ПО поддерживать определенную работоспособность в заданных условиях.

- ***Зрелость, завершенность (maturity).***
  Величина, обратная частоте отказов ПО. Обычно измеряется средним временем работы без сбоев и величиной, обратной вероятности возникновения отказа за данный период времени.
- ***Устойчивость к отказам (fault tolerance)***
  Способность поддерживать заданный уровень работоспособности при отказах и нарушениях правил взаимодействия с окружением.
- ***Способность к восстановлению (recoverability).***
  Способность восстанавливать определенный уровень работоспособности и целостность данных после отказа, необходимые для этого время и ресурсы.

Usability

Удобство использования (usability) или практичность.
Способность ПО быть удобным в обучении и использовании, а также привлекательным для пользователей.

- ***Понятность (understandability).***
  Показатель, обратный к усилиям, которые затрачиваются пользователями на восприятие основных понятий ПО и осознание их применимости для решения своих задач.
- ***Удобство обучения (learnability).***
  Показатель, обратный усилиям, затрачиваемым пользователями на обучение работе с ПО.
- ***Удобство работы (operability).***
  Показатель, обратный усилиям, предпринимаемым пользователями для решения своих задач с помощью ПО.
- ***Привлекательность (attractiveness).***
  Способность ПО быть привлекательным для пользователей. Этот атрибут добавлен в 2001.

Эффективность

Производительность (efficiency) или эффективность =
Способность ПО при заданных условиях обеспечивать необходимую работоспособность по отношению к выделяемым для этого ресурсам. Можно определить ее и как отношение получаемых с помощью ПО результатов к затрачиваемым на это ресурсам всех типов.

- ***Временная эффективность*** *(time behaviour).*

  Способность ПО выдавать ожидаемые результаты, а также обеспечивать передачу необходимого объема данных за отведенное время.

- ***Эффективность использования ресурсов*** *(resource utilisation).*

  Способность решать нужные задачи с использованием определенных объемов ресурсов определенных видов. Имеются в виду такие ресурсы, как оперативная и долговременная память, сетевые соединения, устройства ввода и вывода, и пр.

Переносимость

Переносимость (portability) **-** Способность ПО **сохранять работоспособность при переносе из одного окружения в другое**, включая организационные, аппаратные и программные аспекты окружения.

Иногда эта характеристика называется в русскоязычной литературе мобильностью.

- ***Адаптируемость*** *(adaptability).*

Способность ПО приспосабливаться к различным окружениям без проведения для этого действий, помимо заранее предусмотренных.

- ***Удобство установки*** *или* ***Настраиваемость (installability)*** – способность ПО быть установленным или развернутым в определенном окружении (в специфицированной среде).
- ***Способность к сосуществованию*** *(coexistence).*

Способность ПО сосуществовать с другими программами в общем окружении, деля с ними ресурсы.

- ***Удобство замены*** *(replaceability) другого ПО данным.*

Возможность применения данного ПО вместо других программных систем для решения тех же задач в определенном окружении.

Сопровождаемость

Удобство сопровождения (maintainability).

Удобство проведения всех видов деятельности, связанных с сопровождение программ.

- ***Анализируемость (analyzability) или удобство проведения анализа.***
  Удобство проведения анализа ошибок, дефектов и недостатков, а также удобство анализа необходимости изменений и их возможных последствий.
- ***Удобство внесения изменений (changeability).***
  Показатель, обратный трудозатратам на выполнение необходимых изменений.
- ***Стабильность (stability).***
  Показатель, обратный риску возникновения неожиданных эффектов при внесении необходимых изменений.
- ***Удобство проверки (testability).***
  Показатель, обратный трудозатратам на проведение тестирования и других видов проверки того, что внесенные изменения привели к нужным результатам.

ГОСТ Р ИСО/МЭК 12207

ГОСУДАРСТВЕННЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

«Информационная технология. Процессы жизненного цикла программных средств»

Издание официальное.

РАЗРАБОТАН Всероссийским научно-исследовательским институтом стандартизации (ВНИИстандарт) Госстандарта России

ВНЕСЕН Техническим комитетом по стандартизации ТК 22 «Информационная технология»

ПРИНЯТ И ВВЕДЕН В ДЕЙСТВИЕ Постановлением Госстандарта России от 23 декабря **1999 г.** № 675-ст

ГОСТ Р ИСО/МЭК 12207-2010
Информационная технология. Системная и программная инженерия

Основные процессы (primary life cycle processes) - процессы, которые реализуются «главными» участниками жизненного цикла программного средства.

Поддерживающий процесс **поддерживает другой процесс** как неделимую часть с различными целями и **вносит определенный вклад в успех и качество программного проекта** (Project).

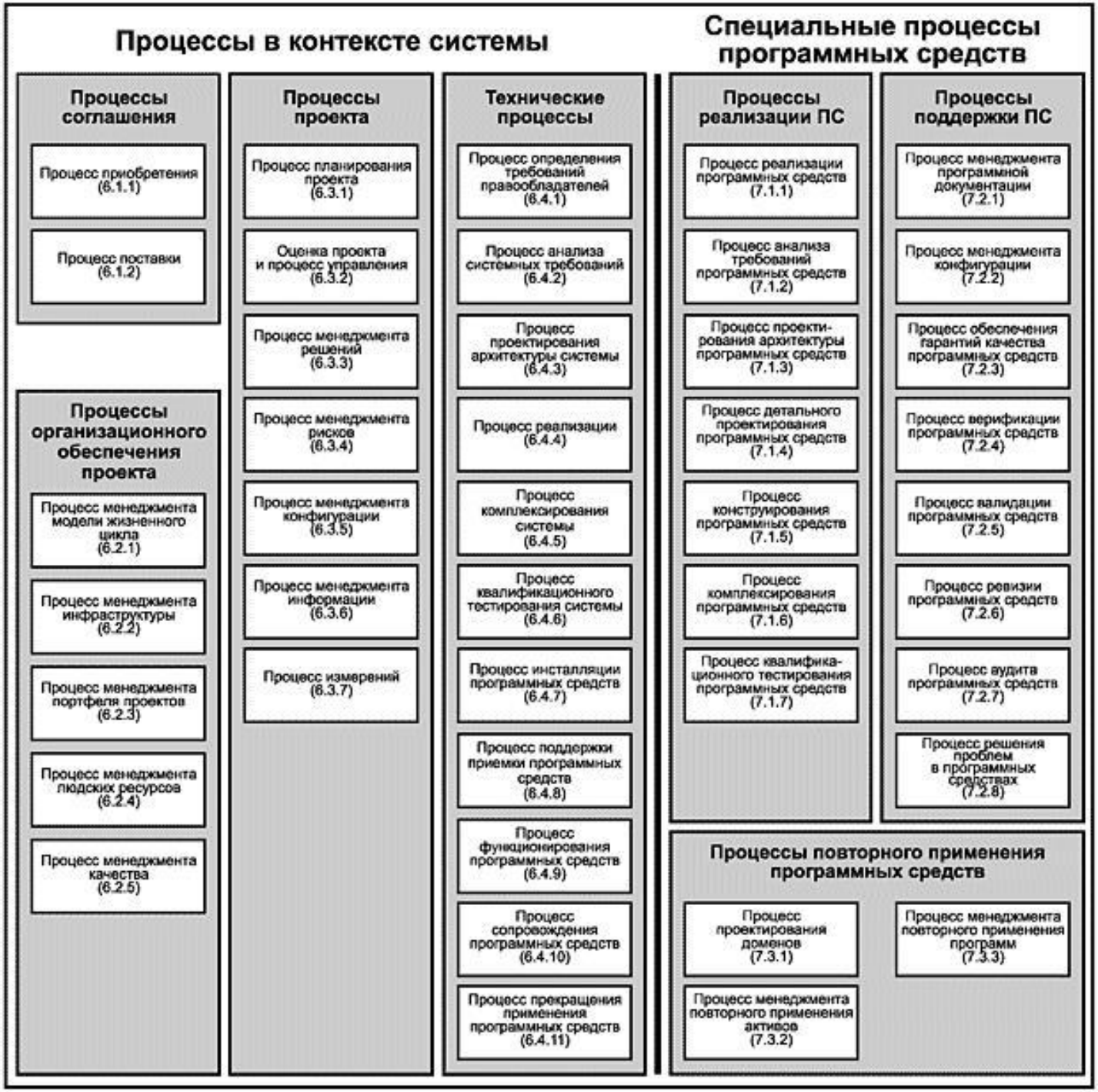
Организационные процессы (organizational life cycle processes) связаны с управлением, созданием необходимой инфраструктуры, обучением персонала, внесением улучшений в процесс.

ГОСТ Р ИСО/МЭК 12207-2010
Информационная технология. Системная и программная инженерия

группирует различные виды деятельности, которые могут выполняться в течение жизненного цикла программных систем, в **семь групп** процессов.

- a) ***процессы соглашения*** - два процесса;
- b) ***процессы организационного обеспечения*** проекта - пять процессов;
- c) ***процессы проекта*** - семь процессов ;
  d) ***технические процессы*** - одиннадцать процессов;
- e) ***процессы реализации программных средств*** - семь процессов ;
- f) ***процессы поддержки программных средств*** - восемь процессов;
- g) ***процессы повторного применения*** программных средств - три процесса .

Группы процессов **жизненного цикла (12207-2010)**



Настоящий стандарт устанавливает строгую связь между *системой* и применяемыми в ней *программными средствами*.

Программное средство трактуется как единая часть общей **системы**, выполняющая определенные функции в данной системе, что осуществляется посредством выделения требований к **программным средствам** из требований к **системе**, проектирования, **производства программных средств и объединения их в систему.**

Этот принцип является фундаментальной предпосылкой для настоящего стандарта, в котором **программные средства всегда существуют в контексте системы,** даже если система состоит из единственного процессора, выполняющего программы.

6.4 Технические процессы

- 6.4.1 Процесс определения требований правообладателей
- 6.4.2 Процесс анализа системных требований
- 6.4.3 Процесс проектирования архитектуры системы
- **6.4.4 Процесс реализации**
  - 6.4.4.1 Цель

    Цель процесса реализации заключается в создании заданных элементов системы.

    Примечание - Пользователи настоящего стандарта могут иметь намерение работать с программными продуктами или программными элементами больших систем. Процесс реализации программных средств (см. 7.1.1) является соответствующим примером процесса реализации в [18], приспособленного к частным потребностям реализации программного продукта или услуги.
- **6.4.5 Процесс комплексирования системы**
- **…**

- Конструкции процессов в ИСО/МЭК 12207 и ИСО/МЭК 15288

ГОСТ Р ИСО/МЭК 12207-2010 Информационная технология. Системная и программная инженерия. Процессы жизненного цикла программных средств

Процессы требуют цели и результатов. Все процессы имеют, как минимум, один вид деятельности. Процессы с их сформулированными целями и выходами составляют эталонную модель процессов (ЭМП). ЭМП представлена в приложении В

Виды деятельности являются структурным компонентом для группирования связанных задач. Виды деятельности предоставляют средства для рассмотрения связанных задач в пределах процесса с целью улучшения понимания и взаимосвязей процессов. Если деятельность достаточно согласована, то она может быть преобразована в процесс более низкого уровня посредством определения цели и совокупности выходов

Задача является детализированным условием реализации процесса. Она может служить требованием (должно»), рекомендацией («следует») или разрешением («может»)

Примечания используются, если появляется необходимость в поясняющей информации для лучшего описания содержания или структуры процесса. Примечания обеспечивают понимание, относящееся к реализации или области применяемости, такой как списки, примеры и другие

Каждый **процесс** описывается в терминах **цели** и желаемых **выходов**, списков **деятельностей** (работ) **и задач**, которые необходимо выполнять для достижения этих результатов.

- **Процесс анализа требований к программным средствам**
- 7.1.2.1 **Цель**
  Цель процесса анализа требований к программным средствам заключается в *установлении требований к программным элементам системы*.

  7.1.2.2 **Выходы**
  a) определяются требования к программным элементам системы и их интерфейсам;
  b) требования к программным средствам анализируются на корректность и тестируемость;
- **…**

7.1.3 Процесс проектирования архитектуры программных средств

- 7.1.3.1 **Цель** процесса проектирования архитектуры программных средств заключается в обеспечении проекта для программных средств, которые реализуются и могут быть верифицированы относительно требований.
- 7.1.3.2 **Выходы**
  - a) разрабатывается проект архитектуры программных средств и устанавливается базовая линия, описывающая программные составные части, которые будут реализовывать требования к программным средствам;
  - b) определяются внутренние и внешние интерфейсы каждой программной составной части;
  - c) устанавливаются согласованность и прослеживаемость между требованиями к программным средствам и программным проектом.

- 7.1.3.3 **Виды деятельности и задачи**
- **7.1.3.3.1 Проектирование архитектуры программных средств**

Для каждого программного элемента вид деятельности состоит из решения следующих **задач:**

7.1.3.3.1.1 ***преобразовать требования к программным составным частям в архитектуру,*** которая описывает верхний уровень его структуры и идентифицирует программные компоненты.

7.1.3.3.1.2 ***разработать и документально оформить проект верхнего уровня для внешних интерфейсов*** программной составной части и интерфейсов между ней и программными компонентами.

7.1.3.3.1.3 разработать и документально ***оформить проект верхнего уровня для базы данных***.

7.1.3.3.1.4 *разработать и документально оформить предварительные версии пользовательской документации*.

7.1.3.3.1.5 определить и документировать требования к предварительному тестированию и график работ по комплексированию программных средств.

7.1.3.3.1.6 оценить архитектуру программной составной части, проекты по интерфейсам и базе данных.7.1.3.3.1.7 проводить ревизии.

7.2.4 Процесс верификации программных средств

- 7.2.4.1 **Цель процесса верификации** программных средств заключается в подтверждении того, что каждые программный рабочий продукт и (или) услуга процесса или проекта должным образом отражают заданные требования.
- 7.2.4.2 Выходы
  В результате успешного осуществления процесса верификации программных средств:
  a) разрабатывается и осуществляется **стратегия в**ерификации;
  b) определяются **критерии** верификации всех необходимых программных рабочих продуктов;
  c) выполняются требуемые **действия** по верификации;
  d) определяются и регистрируются дефекты;
  e) результаты верификации становятся доступными заказчику и другим заинтересованным сторонам.
- 7.2.4.3 Виды деятельности и задачи
  - **7.2.4.3.1 Реализация процесса**
    7.2.4.3.2 **Верификация**

Вид деятельности - верификация

7.2.4.3.2 **Верификация состоит из решения следующих задач:**

- **7.2.4.3.2.1 Верификация требований.** Требования должны быть верифицированы с учетом следующих критериев:
- a) системные требования являются согласованными, выполнимыми и тестируемыми;
- b) системные требования соответственно распределены по техническим, программным элементам и ручным операциям согласно критериям проекта;
- c) требования к программным средствам согласованы, выполнимы, проверяемы и точно отражают системные требования;
- d) требования к программным средствам, связанные с безопасностью, защитой и критичностью, являются корректными, что показано соответствующими строгими методами.
- **7.2.4.3.2.2 Верификация проекта**
- Проект должен быть верифицирован с учетом следующих критериев:
- a) проект корректируется, согласуется с требованиями и обеспечивает прослеживаемость к ним;
- b) проект осуществляет надлежащую последовательность событий, входы, выходы, интерфейсы, логические связи, назначение сроков и размеров финансирования, а также обнаружение ошибок, локализацию и восстановление;
- c) выбранный проект может быть выведен из требований;
- d) проект корректно реализует требования по безопасности, защищенности и другим критическим свойствам, как показано соответствующими строгими методами.
- **7.2.4.3.2.3 Верификация кода**
- **7.2.4.3.2.4 Верификация комплексирования**
- **7.2.4.3.2.5 Верификация документации**

ИАН СОММЕРВИЛЛ

Инженерия программного обеспечения

6-е издание

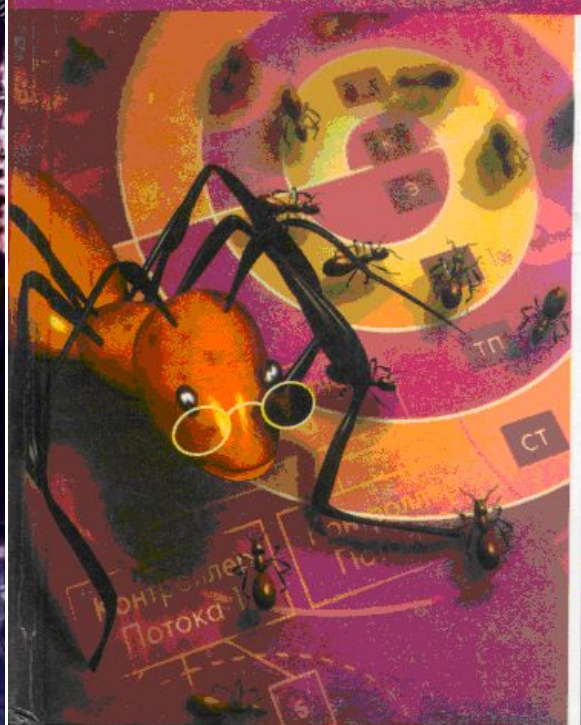
С. А. Орлов

ПИТЕР

ТЕХНОЛОГИИ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

РАЗРАБОТКА СЛОЖНЫХ ПРОГРАММНЫХ СИСТЕМ

УЧЕБНИК ДЛЯ ВУЗОВ



- для студентов и преподавателей высших учебных заведений направлени «Информатика и вычислительная техника»
- фундаментальный курс по программной инженерии

РЕКОМЕНДОВАНО МИНИСТЕ ОБРАЗОВАНИЯ РФ

Факторы, способствующие появлению и развитию SE

- Почему необходимо так **много времени для завершения** разработки программы?
- Почему ее **стоимость столь высока**?
- Почему мы не можем **найти все ошибки** до то, как мы передадим программу нашим заказчикам?

После конференции подкомитета НАТО по науке и технике
(Германия, 50 разработчиков из 11 стран,
термин «программная инженерия»)
состоялась встреча 22-х руководителей projектов (в Лондоне).
где заговорили о надвигающемся кризисе ПО,
предложена концепция жизненного цикла
как последовательности шагов-стадий,
которые необходимо выполнить

Проблемы, относящиеся к ПрОбес

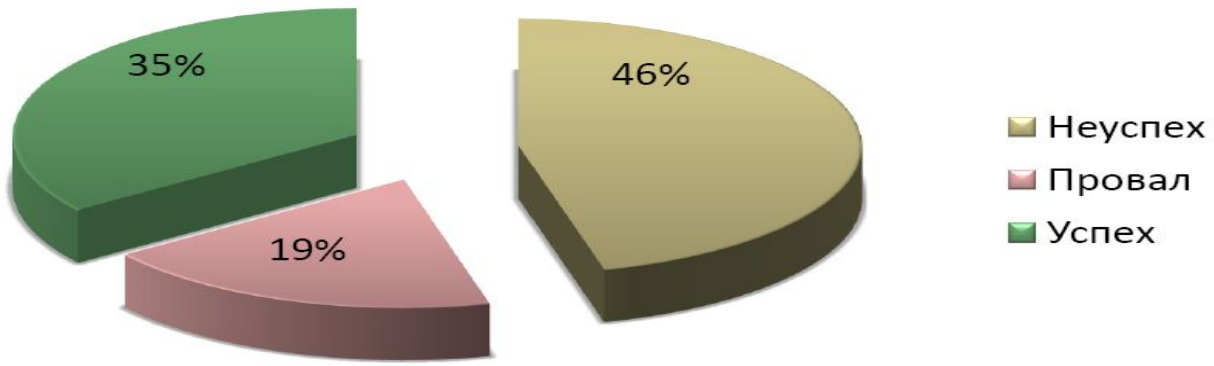
Усилия, затрачиваемые на сопровождение программного обеспечения, начали поглощать ресурсы с тревожащей скоростью.

Стоимость систем многократно возрастала по сравнению с расчетной, системы получались ненадежными и сложными в эксплуатации и сопровождении.

«Кризис программирования": программные проекты (Projects) часто отменяются до своего завершения, а завершенные проекты часто выходят за рамки бюджета и сроков, причем их результаты имеют невысокое качество;

«Программные проекты часто отменяются до своего завершения, а завершенные проекты часто выходят за рамки бюджета и сроков, причем их результаты имеют невысокое качество.» [Кони Бюрер (Rational Software)].

Статистика производства ПО

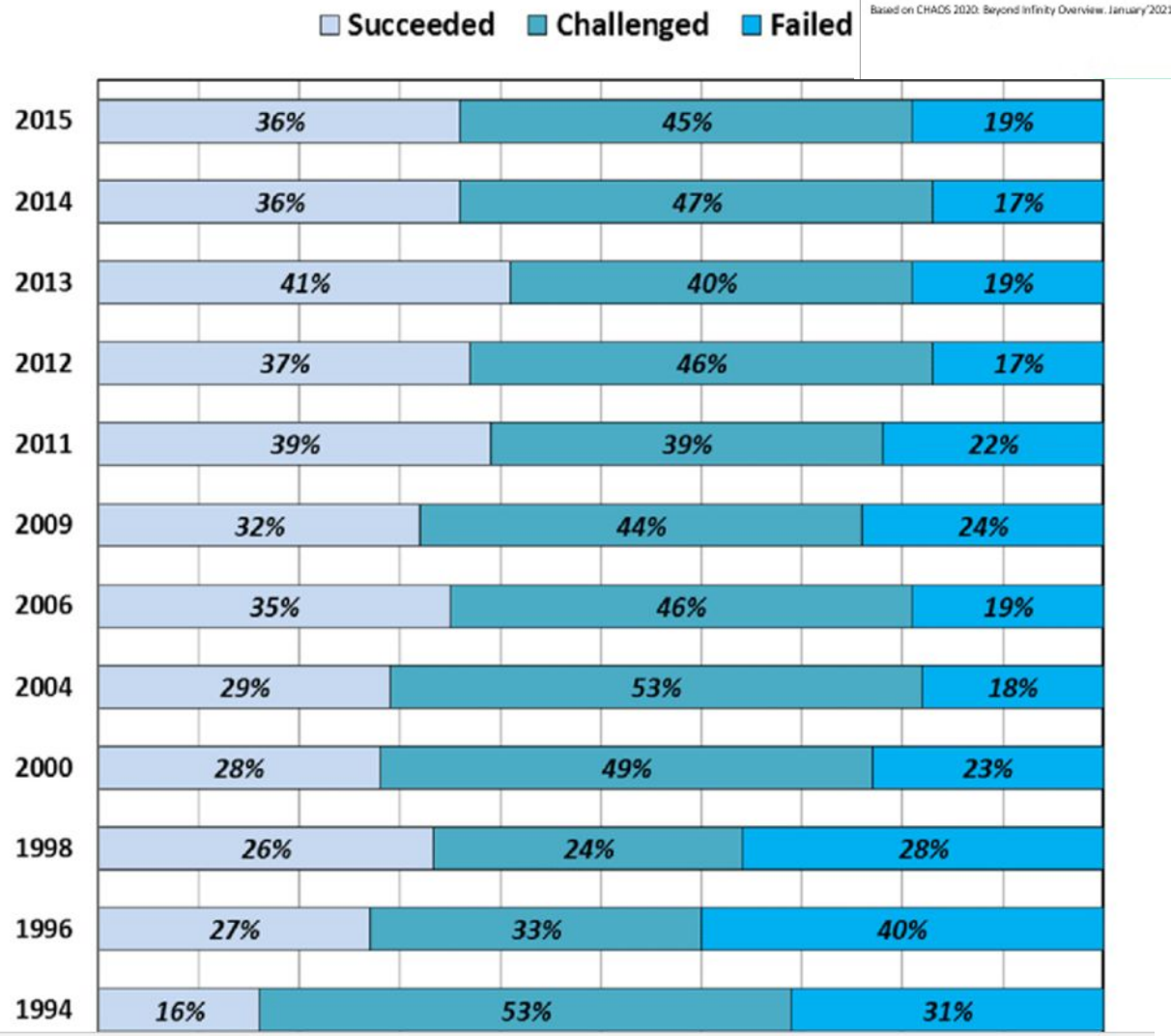


Источник: "The Chaos Report" The Standish Group, 2008

- CHAOS research encompasses 18 years of data on why projects succeed or fa representing **more than 90,000 completed IT projects.** However, for our nev database we eliminated cases from 1994 though 2002, since they did not matc the current requirements for analysis. The new database has **just under 50,00 projects.**
- CHAOS results provide a global view of project statistics but do tend to have heavier concentration on the United States and Europe. For each reporting period, about 60% of the projects are U.S. based, 25% are European, and the remaining 15% represent the rest of the world.

Standish Group Chaos Report 2020

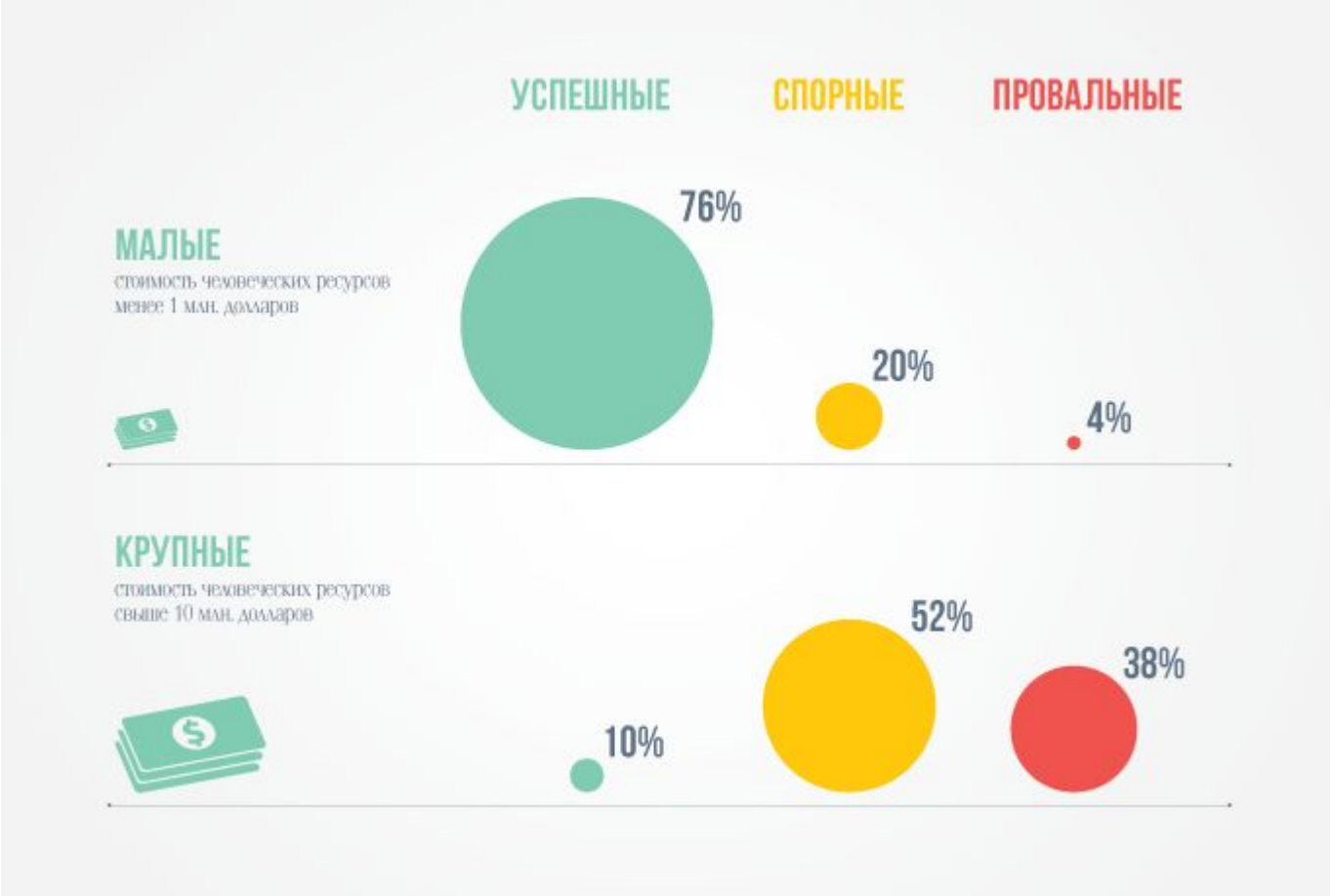
из Henny Portman's Blog:



Зависимость успеха от масштаба проекта

Статистика The Standish Group демонстрирует зависимость успеха от масштаба проекта:

По небольшому проекту прогноз сроков окажется более достоверным [The Standish Group. CHAOS MANIFESTO 2013]

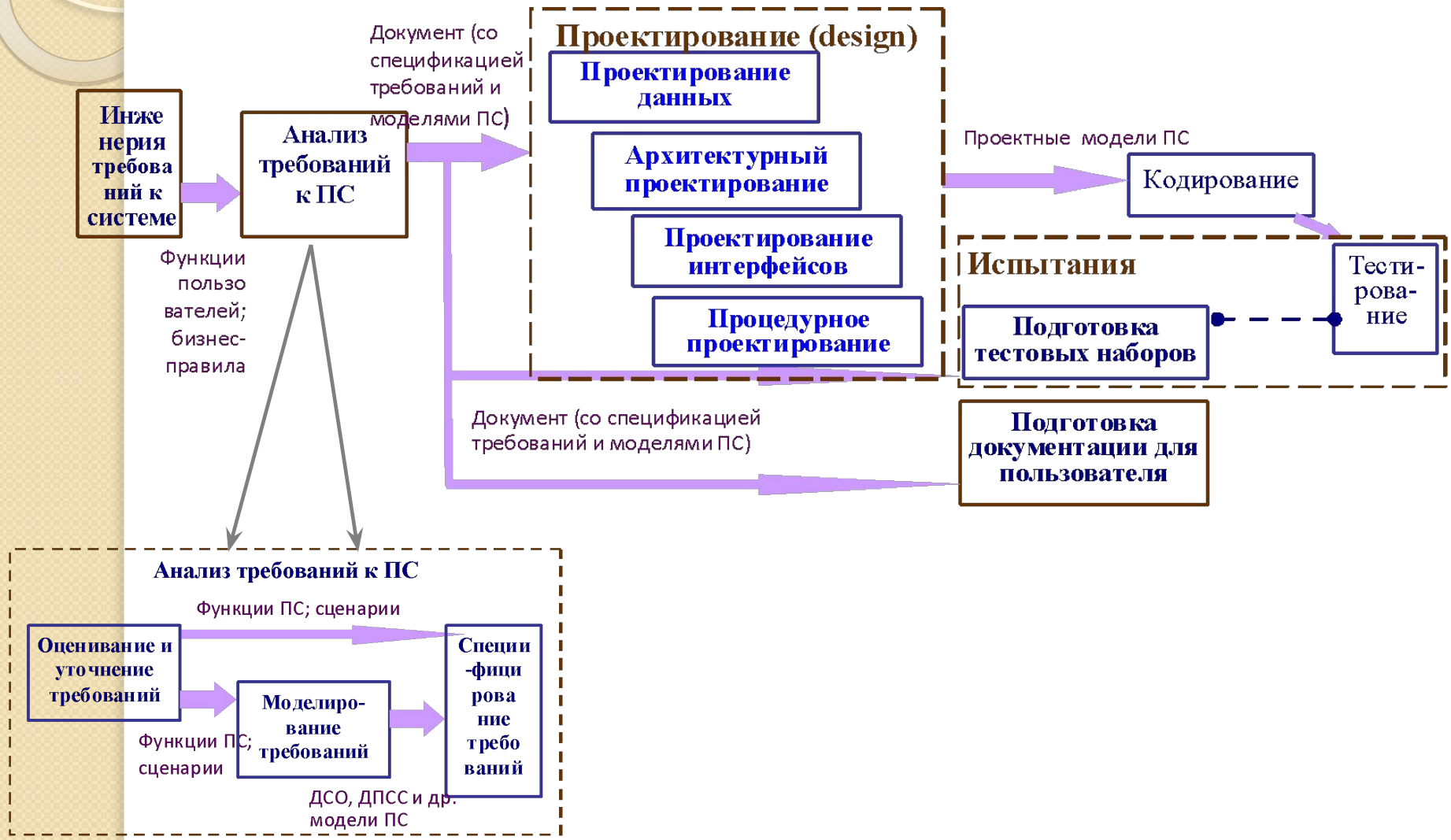


Технология программирования – это применение систематического, упорядоченного, поддающегося количественному определению подхода для разработки программного обеспечения, работы с ним и его сопровождения, то есть, применение инженерного дела к программному обеспечению[ANSI/IEEE 610.12-1990]

Понятие процесса

- *Технологический* ***процесс*** ***(process) - элемент в структуре ЖЦ ПО \ПС —*** *множество взаимосвязанных видов деятельности****, решающих некоторую общую задачу (problem)*** *или связанную совокупность задач* (напр, *анализ требований*, или *процесс обеспечения качества)*
- ***Деятельность*** ***(activity)*** - часть *процесса*, выполняемая одним человеком или группой, ***преобразующими получаемую ими входную информацию в выходную.*** Каждая **деятельность** представляется набором инженерных рабочих задач.
- ***Задачи*** ***(Tasks)*** атомарны (выполняются обычно одним человеком) и состоят в выполнении некоторого действия.

Процессы и деятельности при разработке ПО



Деятельность «Аттестация»

состоит из следующих ***задач:***

***Подготовка выбранных требований к испытаниям*, контрольных примеров** и технических условий испытаний **к анализу результатов испытаний**.

Проведение испытаний, включая:

- a) **испытания при критических, граничных и особых значениях исходных данных**;
- b) **испытание программного продукта на способность изолировать и минимизировать эффект ошибок** с постепенным понижением влияния сбоев и запросом помощи оператора при критических, граничных и особых условиях;
- c) **испытание при участии репрезентативно выбранных пользователей**, могущих успешно решать свои задачи при использовании данного программного продукта.

***Подтверждение того, что программный продукт удовлетворяет* заявленным возможностям.**

Общая схема процессов (common process framework)

устанавливается определением небольшого числа ***видов деятельности*** (процессов), которые применимы ко всем проектам ПО, независимо от их размера или сложности.

- **Системная/информационная инженерия**
- **Анализ требований к программному обеспечению**
- **Проектирование**
- **Создание кода программного обеспечения**
- **Испытания**
- **Сопровождение программного средства**

- **Фазы определения** (**какие** функции, информация, «поведение»…),
- **разработки** (**как** построить) и
- **сопровождения** (исправление, адаптация, модернизация, предотвращение)

Пропорции стадий этапа разработки

*[*В. Иванов, «Руководство по управлению внедренческими **проектами на базе MS PROJECT 2000** и рекомендаций PMI»)*] Этап разработки разделяется на стадии :*

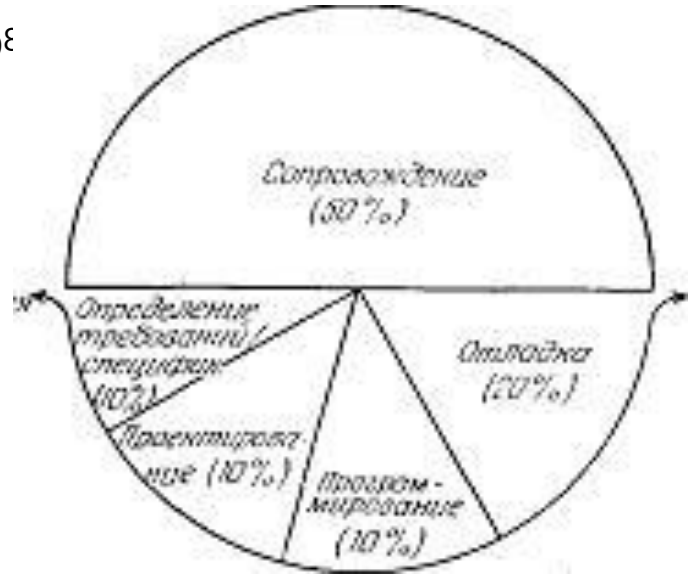
- *-**Постановка - 34%***
- ***-Кодирование - 21%***
- ***-Тестирование - 45%***

Брукс: *В течение ряда лет … пользуюсь следующим эмпирическим правилом:*

- ***1/3 - планирование,***
- *1/6 - написание программ,*
- ***1/4 - тестирование** компонентов и предварительное системное тестирование,*
- ***1/4 - системное тестирование** при наличии всех компонентов.*

(По материалам **Р.Гласс, Р.Нуазо**, "Сопровождение ПРО«, 198

- *Определение и специфик-я требов-й – 10%;*
- ***Проектирование – 10%;***
- *Программирование – 10%;*
- ***Отладка – 20%;***
- ***Сопровождение – 50%.***



(Р. Боровко / CNews Analytics): «Согласно отраслевым исследованиям, компании, разрабатывающие ПО, тратят 38% своих времени и денег на исправление уже написанного кода…»

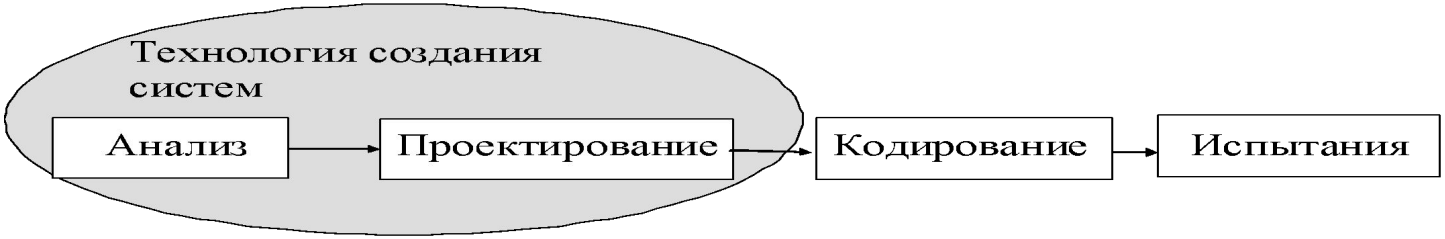
Жизненный цикл программного средства

Жизненный цикл программной системы представляет собой непрерывный процесс, начинающийся с момента принятия решения о ее создании и заканчивается в момент полного изъятия ее из эксплуатации.

Под ***моделью жизненного цикла*** понимается **схема**структура, определяющая последовательность выполнения **деятельностей, связанных с разработкой, эксплуатацией и сопровождением ПО** и их взаимосвязи (на протяжении жизненного цикла).

Стандарт ISO/IEC 12207 *не предопределяет конкретной модели жизненного цикла или метода разработки программного средства. Пользователи, применяющие настоящий стандарт, должны сами выбирать модель жизненного цикла применительно к своему программному проекту и распределять процессы, работы и задачи, выбранные из настоящего стандарта, на данной модели.*

Линейные модели



Линейная последовательная модель.

Каскадная модель (*однократный проход* , ***водопадная стратегия***, *waterfall model*) - **систематический подход к разработке программного обеспечения**, который начинается на системном уровне и проходит через анализ, проектирование, кодирование, испытания и сопровождение.

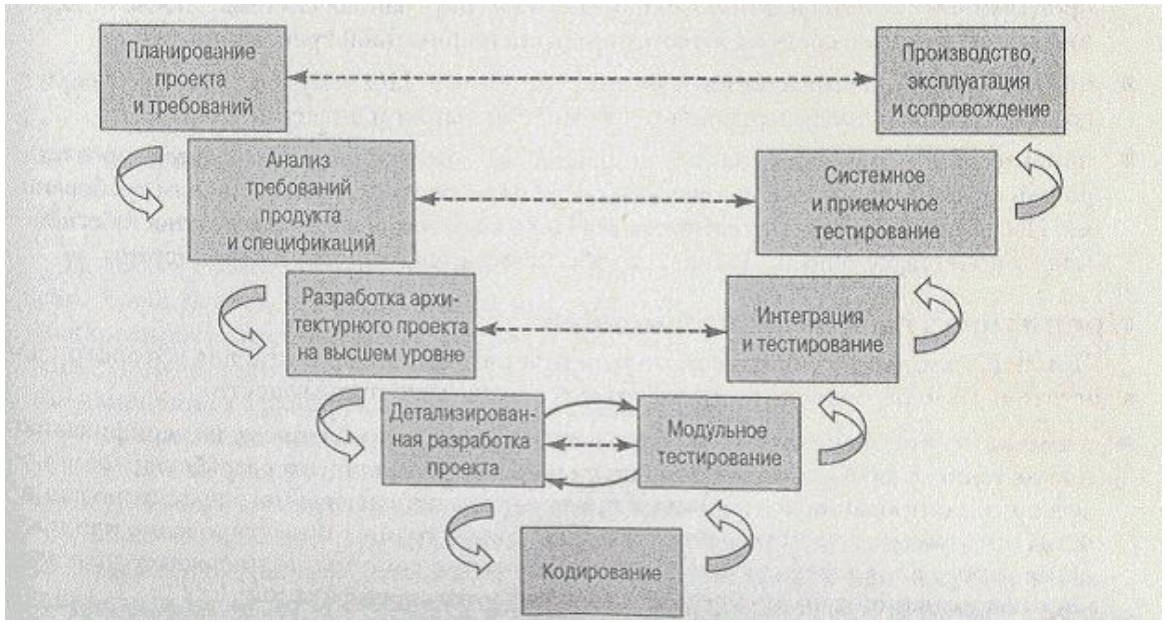
- 1970 -1985 гг

Элементы конфигурации, создаваемые при использовании каскадной модели

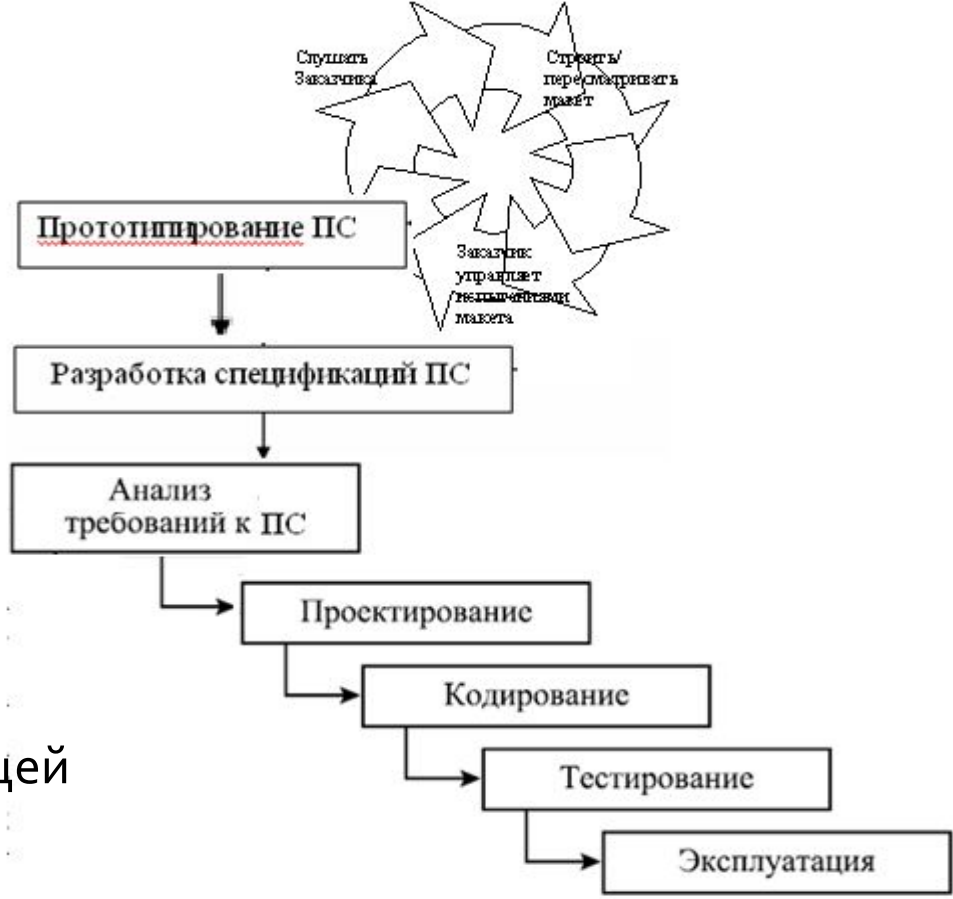
Вид деятельности	Элементы конфигурации (документы)
Анализ требований	Отчет по исследованию реализуемости Предварительные требования
Определение требований	Документ с требованиями
Спецификация системы	Спецификация функций План приемочных испытаний Предварительное руководство пользователя
Архитектурное проектирование	Спецификация архитектуры План испытаний системы
Проектирование интерфейса	Спецификация интерфейса План испытаний при комплексировании
Детальное проектирование	Спецификация проекта План испытаний программных единиц
Кодирование	Код программы
Испытание программных единиц	Отчет об испытаниях программных единиц
Испытания при комплексировании	Отчет об испытании при комплексировании Окончательное руководство пользователя
Испытание системы	Отчет об испытаниях системы
Приемочные испытания	Окончательный вариант системы и системная документация

V-образная модель жизненного цикла

Цель: **помочь** работающей над проектом команде **в планировании с обеспечением дальнейшей возможности тестирования системы**. В этой модели особое значение придается **действиям, направленным на верификацию и аттестацию продукта**.



Линейные модели.
Модель с прототипированием



Цель: **поэтапное уточнение требований заказчика и получение законченной спецификации,** определяющей разрабатываемую систему.

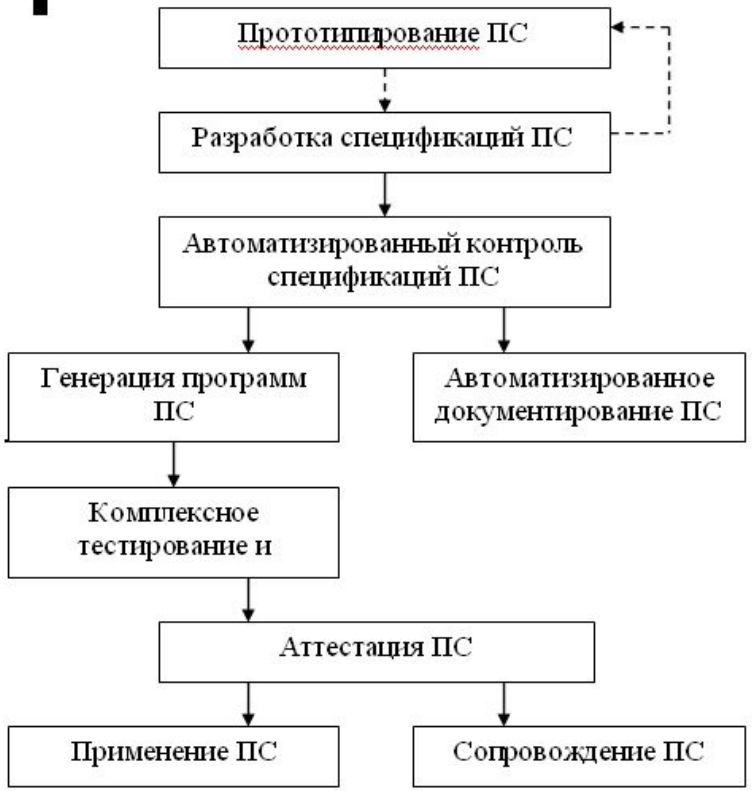
Прототип строится, чтобы служить механизмом для определения требований.

Часто заказчик определяет ***множество общих целей*** для программного средства, но не идентифицирует детально входные данные, их обработку или требования к выходным данным. В других случаях разработчик может быть не уверен в эффективности алгоритмов, пригодности операционной системы или в форме, которую должно бы иметь человеко-машинное взаимодействие.

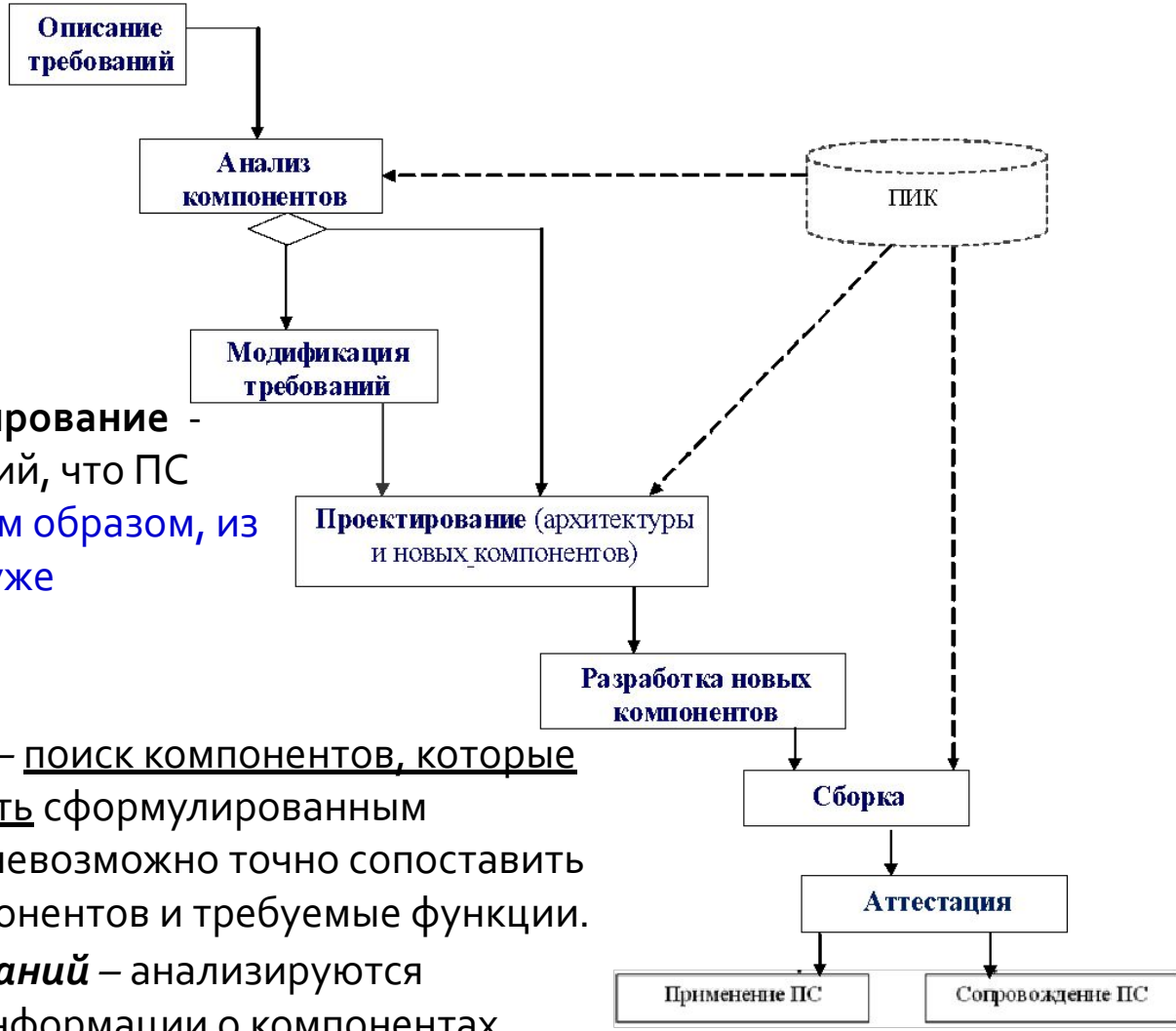
Линейные модели

Компьютерная ТРПО: используются программные инструменты для разработки формализованных спецификаций программ с последующей автоматической генерацией программ и документов.

Это разработка с использованием CASE-средств, **позволяющих разработчику «специфицировать желаемый результат**, а не действия, необходимые для достижения этого результата». Поддерживающее ПС транслирует эти спецификации результата в машинно-исполняемую программу.



Сборочная технология разработки



Сборочное программирование - подход, предполагающий, что ПС конструируется, главным образом, из компонентов, которые уже существуют.

- ***Анализ компонентов*** – поиск компонентов, которые могли бы удовлетворять сформулированным требованиям, обычно невозможно точно сопоставить функции готовых компонентов и требуемые функции.
- ***Модификация требований*** – анализируются требования с учетом информации о компонентах, полученной на предыдущем этапе. Они модифицируются так, чтобы максимально использовать возможности отобранных компонентов.
- ***Проектирование*** – проектируется структура системы с учетом функциональных возможностей отобранных готовых программных компонентов.

Эволюционные модели программных процессов

- **Требования** к бизнесу и продуктам часто **изменяются**,
- **сжатые рыночные сроки** делают завершение всестороннего продукта невозможным.

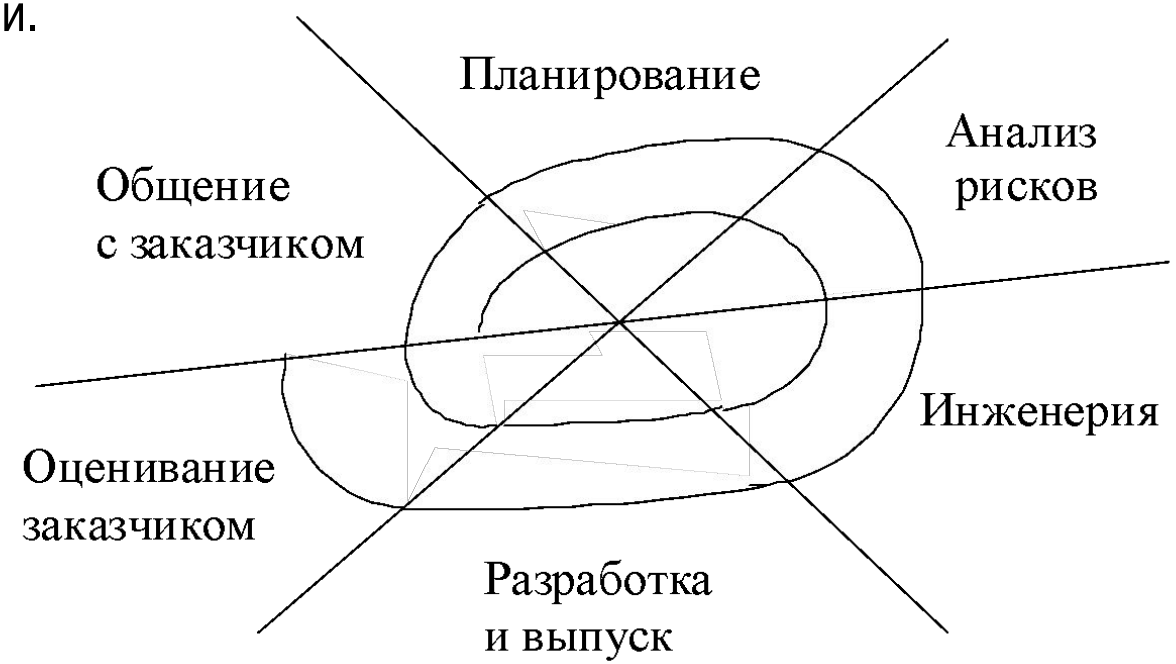
Инженеры ПО нуждаются в *эволюционной модели процессов* - циклическом повторении практически всех фаз.

Эволюционная стратегия: система строится в виде последовательности версий.

Спиральная модель процессов

Модель предусматривает циклическое **повторение практически всех этапов работ.**

При ранних итерациях уточняются спецификации продукта, «выпуск» может быть моделью на бумаге или прототипом. **При поздних** итерациях выпускаются все более увеличивающиеся по сложности версии сконструированной системы, добавляются новые возможности и функции.



Типичная спиральная модель.

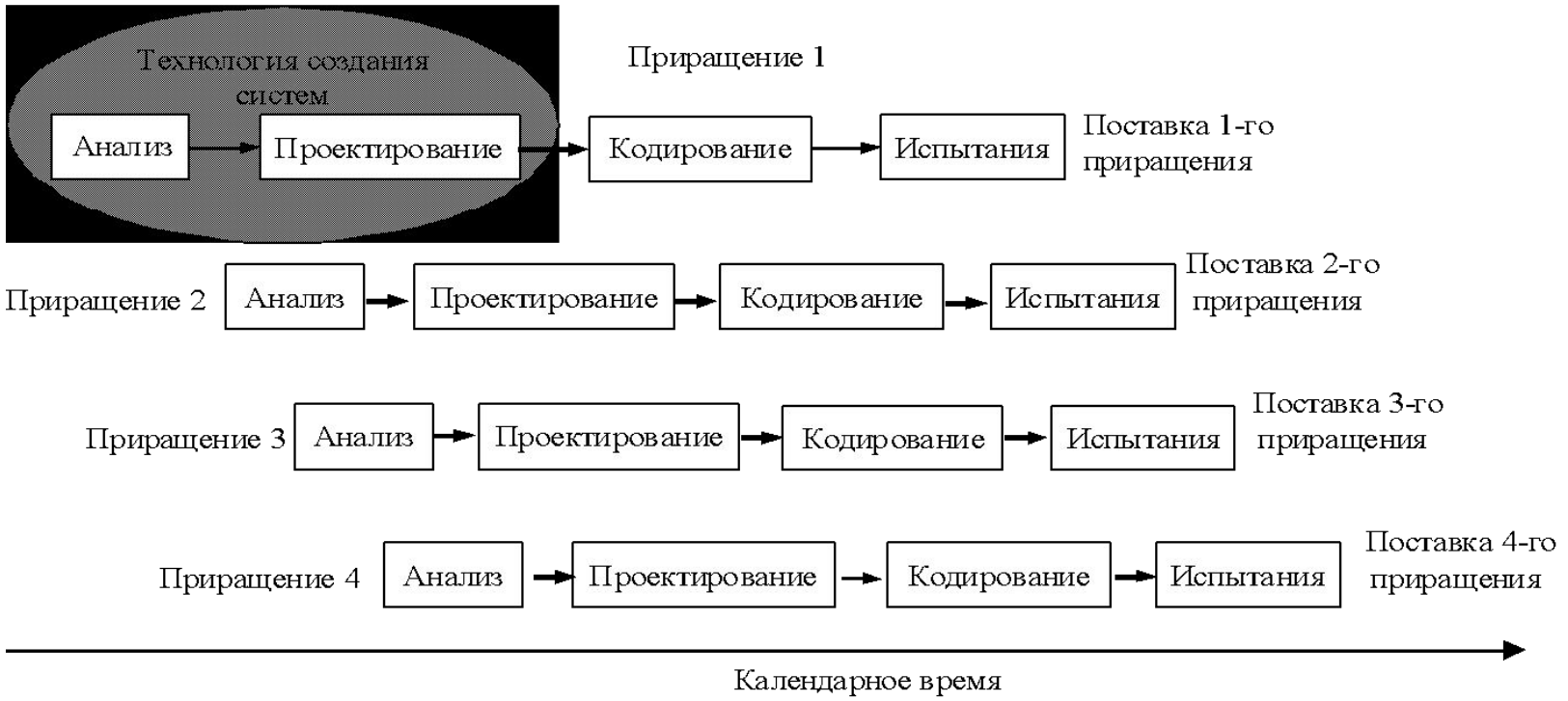
Модель разработки приращениями

...когда стесненные сроки могут помешать реализации всех требований ...

инкрементная стратегия: в начале процесса определяются все пользовательские и системные требования, оставшаяся часть конструирования выполняется в виде последовательности версий.

Изначально должны быть установлены *приоритеты требований.*

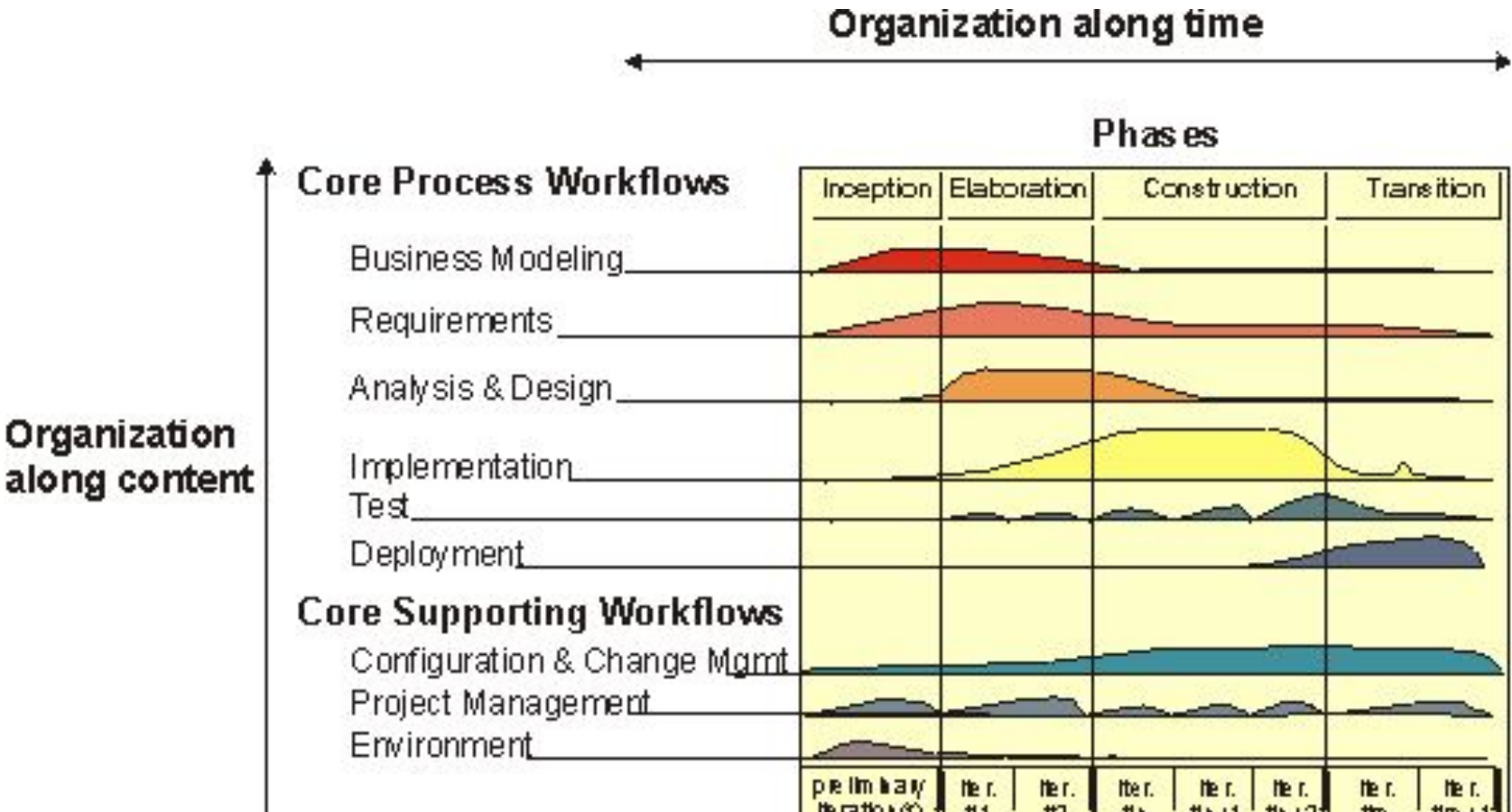
Первая версия реализует часть запланированных возможностей. *В конце каждой* итерации должна получаться работающая *версия продукта*, но с неполной функциональностью.



Модель разработки приращениями.

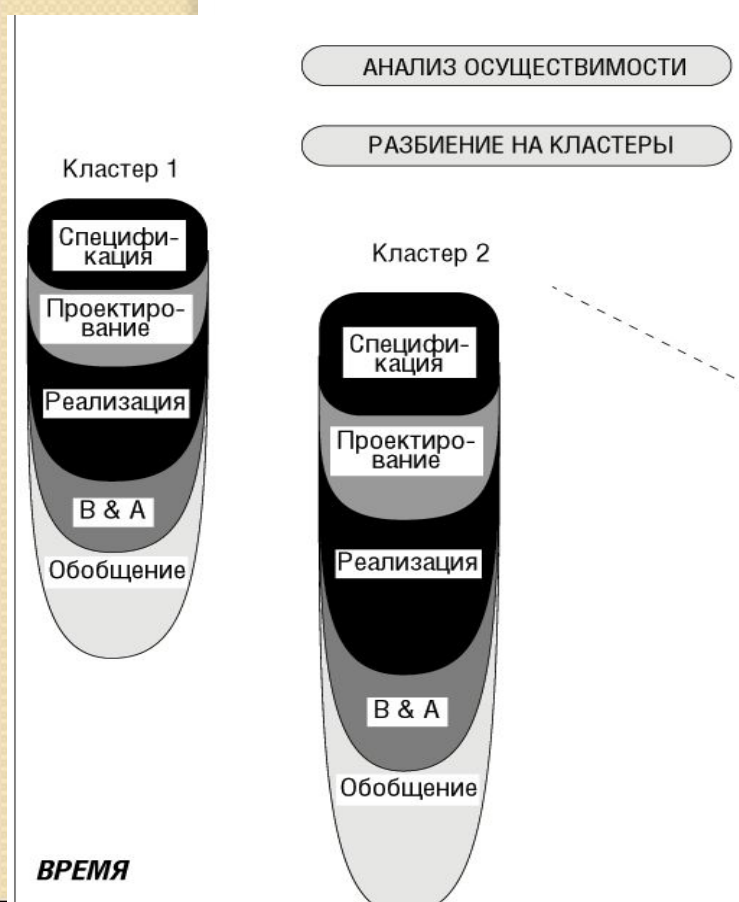
Рациональный унифицированный процесс (RUP) - итеративный процесс, предполагающий разделение проекта на несколько «***мелких проектов***:

- ***Исследование (Inception)***
- ***Уточнение\Проработка плана (Elaboration)***
- ***Построение (Construction)***
- ***Развертывание\«передача» (Transition)***



«КЛАСТЕРНАЯ» МОДЕЛЬ ЖИЗНЕННОГО ЦИКЛА системы

После того как будет определен интерфейс между подсистемами, их дальнейшее проектирование можно вести независимо.



В начале Проекта необходимо пройти через две фазы:

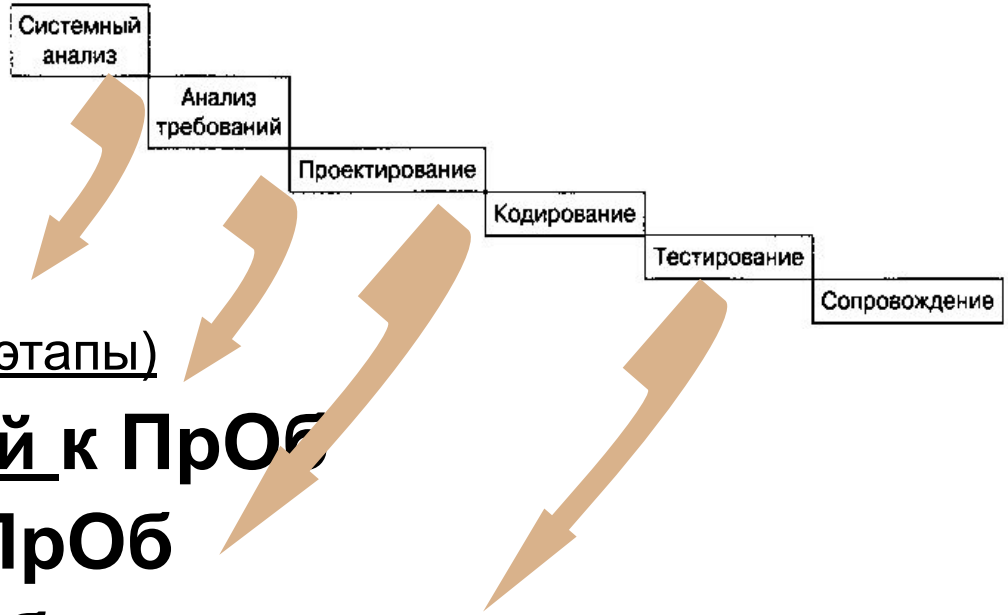
- **анализ осуществимости (feasibility study)**, на основе которого принимается решение о начале работы над Проектом;
- **разбиение Проекта на** небольшое количество параллельных «кластеров» (подгрупп исполнителей) и последовательные процессы производства относительно **независимых подсистем.**

Вертикальная ось представляет последовательную «составляющую» процесса: чем ниже размещена та или иная работа, тем позже она будет выполнена.

Горизонтальное направление отражает параллельную разработку: **задачи на одном уровне могут выполняться в одно время.**

План изучения дисциплины

- Процессы программного обеспечения



- **Системная инже**

  нерия (начальные этапы)
- **Анализ требований к ПрОб**
- **Проектирование ПрОб**
- **Кодирование ПрОбесп и Испытания**
- **Управляющие и поддерживающие процессы**

«Слои» технологии программирования

Методы *ТРПО* представляют собой **«структурные решения», предназначенные для разработки высококачественного ПО эффективным способом**, они задают техническое описание того, "как сделать", чтобы построить программное средство или его компонент. Они включают в себя ***правила моделирования, формализованную нотацию***, другие наглядные приемы, а также способы управления процессом создания ПО.

Слои технологии программирования.

Средства (tools) *ТРПО* обеспечивают автоматическую или полуавтоматическую поддержку процессов и методов. Это ***системы программирования, инструментальные программные средства*** (toolkits, CASE-средства), аппаратура (компьютеры, сети, ...)

CASE - *автоматизированная* или *поддерживаемая ЭВМ технология программирования* (computer-aided software engineering) - программные системы для автоматизации процесса создания ПО, в которых информация, создаваемая одним из средств, используется другим.

- 5-6 чел
- Модель качества

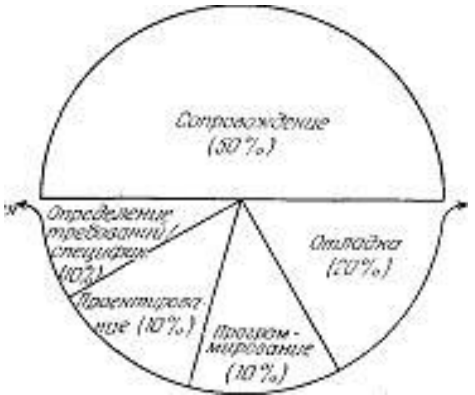
- Пример («своих») ПС разных классов,
  - Какие хар-ки качества

- Система
- системное требование

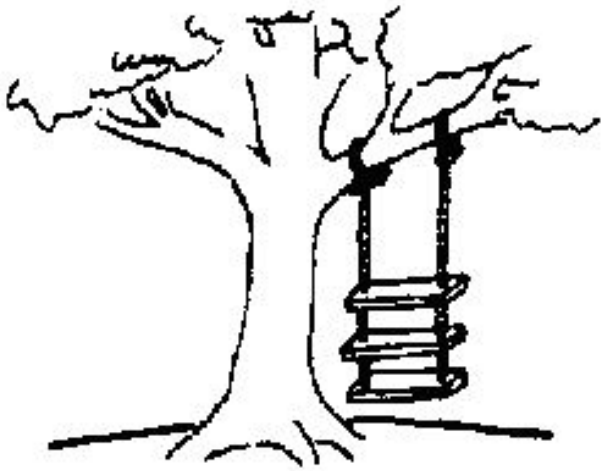
Чем отличается от этих

ЖЦ коллективной разработки Вашего Project?

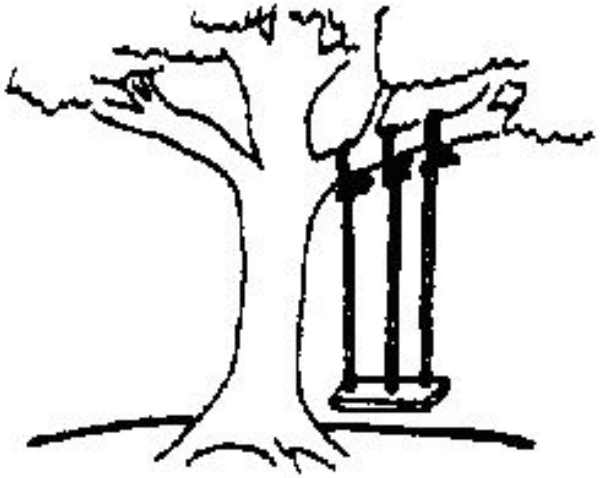
Какой из этапов был наиболее трудоемким?



Как обычно пишутся программы

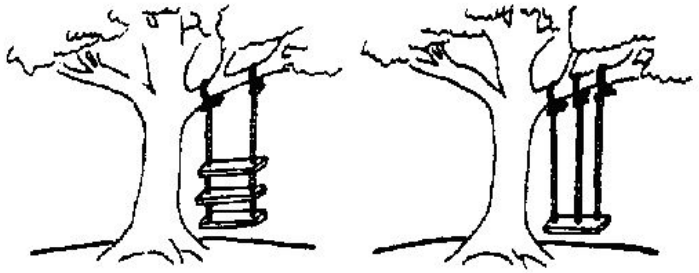


ТАК БЫЛО ПОСТАВЛЕНО ТЕХНИЧЕСКОЕ ЗАДАНИЕ



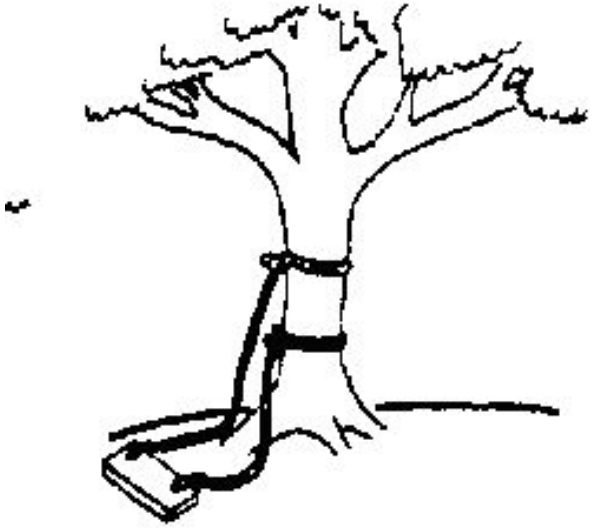
ТАК ЕГО ПОНЯЛИ РАЗРАБОТЧИКИ

Как обычно пишутся программы

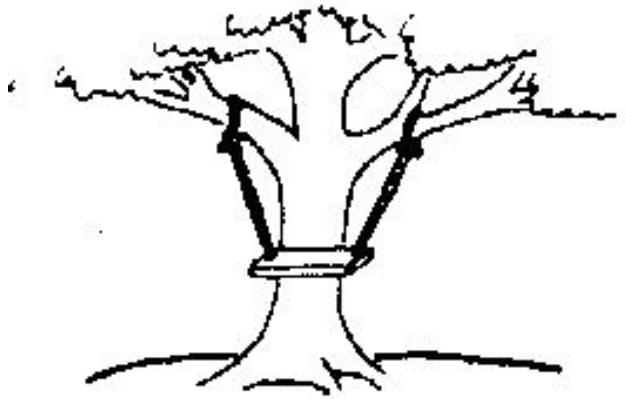


ТАК БЫЛО ПОСТАВЛЕНО ТЕХНИЧЕСКОЕ ЗАДАНИЕ

ТАК ЕГО ПОНЯЛИ РАЗРАБОТЧИКИ

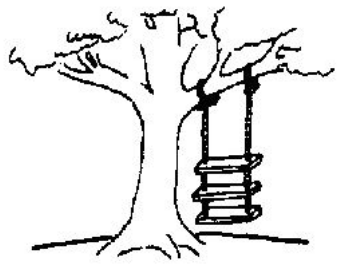


ТАК ЭТУ ЗАДАЧУ И РЕШАЛИ РАНЬШЕ

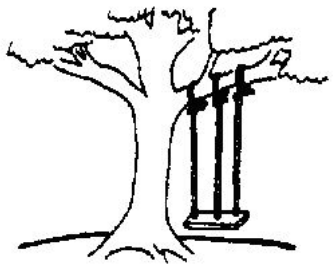


ТАК ЕЕ РЕШИЛИ ТЕПЕРЬ

Как обычно пишутся программы

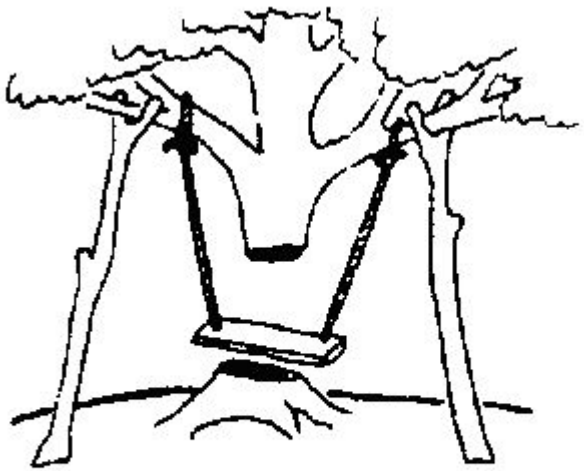


ТАК БЫЛО ПОСТАВЛЕНО ТЕХНИЧЕСКОЕ ЗАДАНИЕ

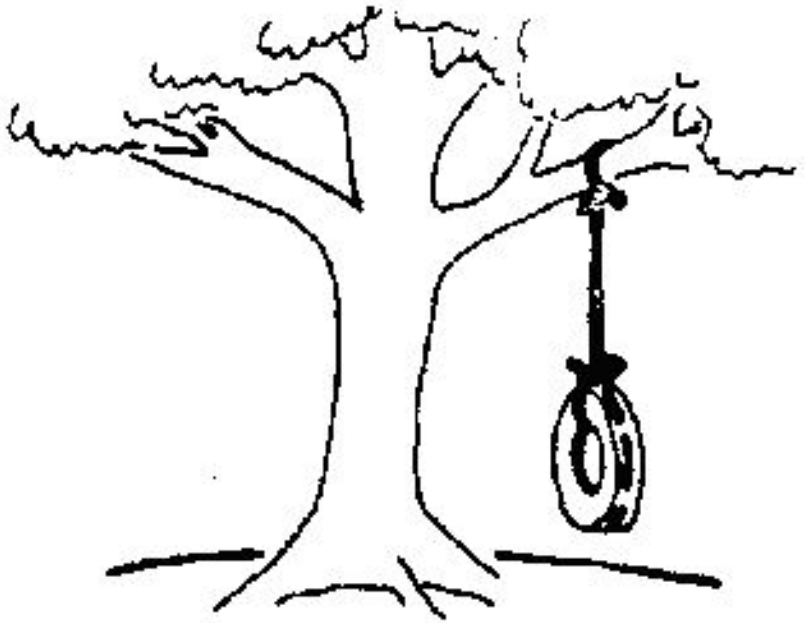


ТАК ЕГО ПОНЯЛ РАЗРАБОТЧИК

ТАК ЕЕ РЕШИЛИ ТЕПЕРЬ



ТАКОЙ ПРОГРАММА СТАЛА ПОСЛЕ ОТЛАДКИ



А, СОБСТВЕННО, ТАК ЕЕ ПРЕДСТАВЛЯЛ СЕБЕ ЗАКАЗЧИК