



Программирование

2 семестр

Осипов Никита Алексеевич
naosipov@itmo.ru

Распределение учебного времени

Семестр	Вид контроля (экз./диф.зач./ зач.)	Контактная работа, час.	Занятий лекц. типа, час.	Лаборат. занятий, час.	Практич. занятий, час.	СРО, час.
2	Экзамен	52,80	16	16	16	55,2

Включает
выполнение
домашнего задания



Содержание дисциплины

4. Реализация технологии объектно-ориентированного программирования

4.1. Классы и объекты.

4.2. Отношения между классами.

4.3. Абстрактные базовые классы. Множественное наследование.

4.4. Метaprogramмирование.

4.5. Сетевое взаимодействие.

4.6. Асинхронное программирование.

4.7. Применение паттернов проектирования при реализации программ.



Содержание дисциплины

5. Взаимодействие с СУБД

5.1. Создание приложений с базой данных SQLite.

5.2. Применение ORM.

6. Создание GUI-приложений

6.1. Создание GUI на основе библиотеки Tkinter.

6.2. Применение ООП при создании компонентов.



- Руководство по разработке качественных требований к программному обеспечению.
- Описаны приемы выявления, формулирования, разработки, проверки, утверждения и тестирования требований, которые помогут создать эффективное ПО.
- Издание дополнено новыми приемами, посвященными разработке требований в проектах гибкой разработки (agile).

Crystal Collection для профессиональных разработчиков программного обеспечения
Научный руководитель проекта – Алистер Коберн

Современные методы описания функциональных требований к системам



Алистер Коберн

- Руководство по написанию вариантов использования.
- В книге представлены начальная, промежуточная и развитая концепции
- Подходит читателям с разным уровнем подготовки.
- Инструкции подкреплены наглядными примерами и упражнениями.

Программный продукт

- Общие программные продукты – автономные программные системы, которые созданы для продажи на открытом рынке программных продуктов любому потребителю
 - Примеры: системы управления базами данных, текстовые процессоры, графические пакеты, средства управления проектами, планировщики задач и т.п.
 - Распространяются как обычный товар «в коробке» или по Интернету
 - Клиент – массовый потребитель
 - Стандартные операции установки на машину клиента

Программный продукт

- Программные продукты, созданные на заказ - программные системы, которые создаются по заказу определенного потребителя
 - Примеры: системы поддержки определенных производственных или бизнес-процессов и т.п.
 - Разрабатываются специально для данного потребителя согласно заключенному контракту
 - Распространяются как внедрение в аппаратно-программной системе клиента – «решение»
 - Клиент – «заказчик» - организация (в широком смысле) со своими особыми требованиями
 - Развертывание и установка с учетом особенностей клиента

Разработка программного обеспечения

- Хаотическая деятельность – "code and fix" ("пишем и правим")
 - единого плана не существует
 - общий проект представляет собой просто смесь краткосрочных решений

Для сложных систем нужна некая организация работы по созданию программного обеспечения

Технология разработки программного обеспечения – комплекс организационных мер, операций и приемов, направленных на разработку программных продуктов высокого качества в рамках отведенного бюджета и в срок

Методики разработки ПО различаются используемой моделью жизненного цикла ПО и уровнем формализма при его создании

Разработка программного обеспечения

- Хаотическая деятельность – "code and fix" ("пишем и правим")
 - единого плана не существует
 - общий проект представляет собой просто смесь краткосрочных решений
- Технология превращает создание программного продукта в упорядоченный процесс
 - работа программиста более прогнозируемая и эффективна
 - создается детальное описание процесса создания системы, особое место в котором занимает планирование (аналогично другим инженерным дисциплинам)
- Облегчённые (lightweight) или гибкие (agile) технологии
 - меньшая ориентация на документацию
 - ориентированность на код

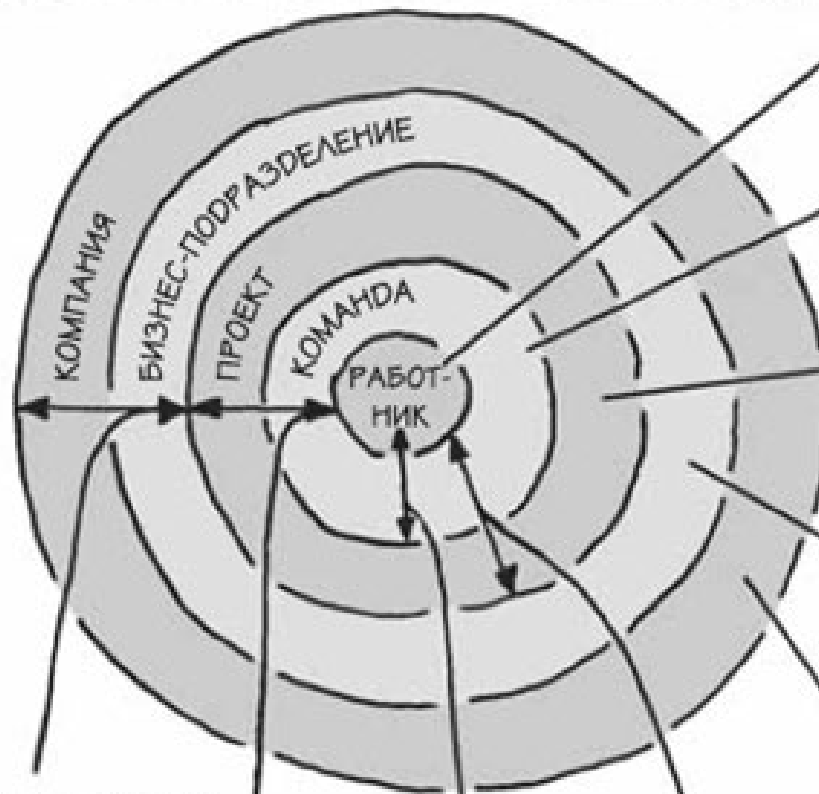
Командная разработка

- Работа в команде является необходимым условием успешности проекта
- Умение работать в команде – важное качество высококвалифицированного разработчика программного обеспечения

Уровни команд

- Отдельный программный компонент, входящий в более крупный проект;
- Компонент должен войти в более общий продукт.
- Разработка нескольких проектов одновременно

УРОВНИ ГРУППОВОЙ РАБОТЫ



Динамика организации
(верхний уровень
групповой работы)

Динамика группы
(нижний уровень
групповой работы)

Взаимодействие
внутри команды

Взаимодействие
между командами

Требует:

- умения разрабатывать ПО
- умения мыслить
- обучения
- мотивации

Вводит:

- социальные навыки
- динамику внутри команды

Объединяет несколько маленьких команд, вводя динамику между ними; большие усилия по координации.

Особого внимания требуют:

- обмен данными
- планирование
- управление ресурсами

Каждый проект должен решать задачи бизнес-подразделения.

На этом уровне на работу влияют:

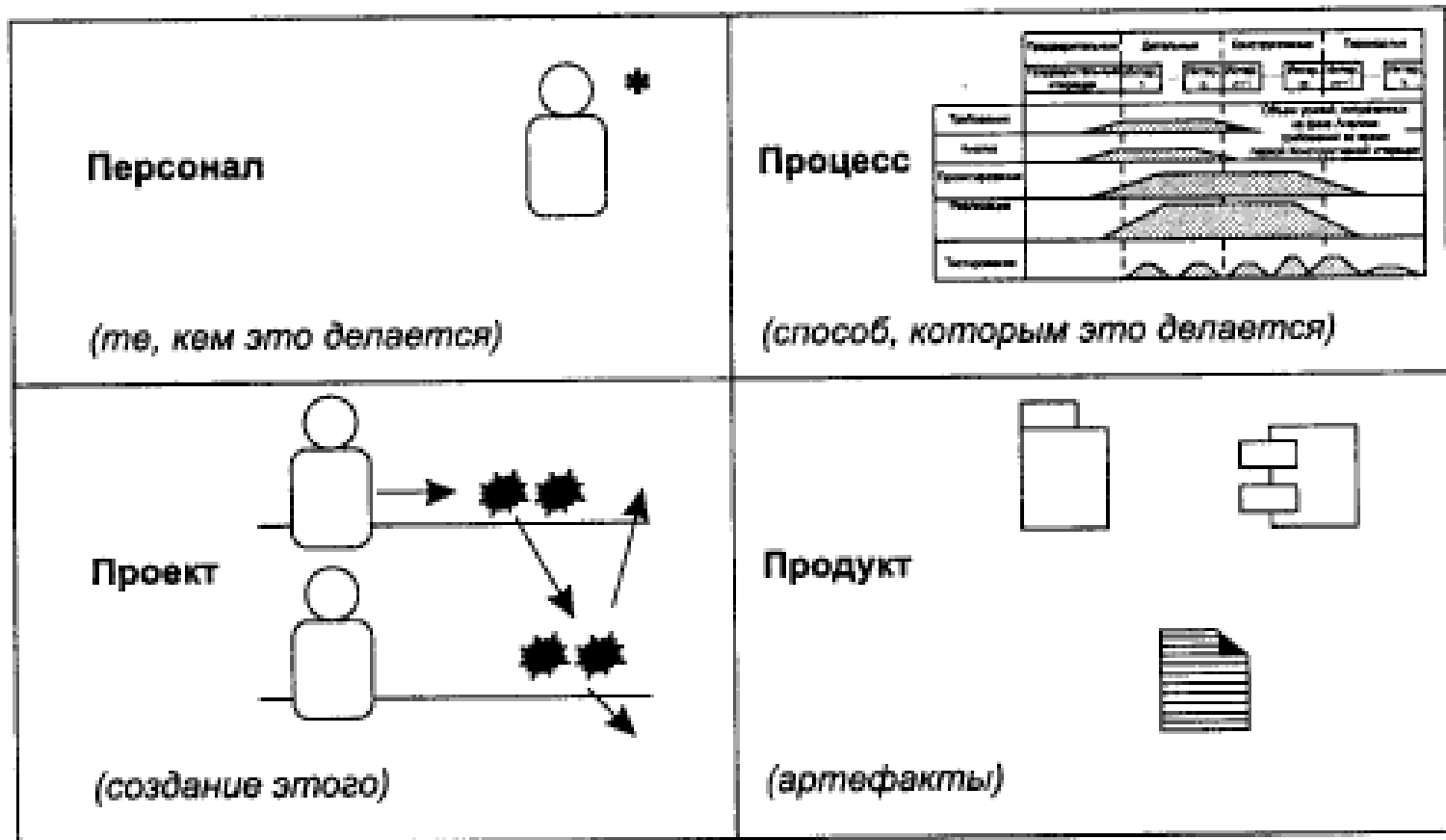
- политика корпорации
- культура
- процедуры

Корпоративный контекст:

- взаимодействие с другими компаниями (клиентами, поставщиками)
- бизнес-стратегия
- схемы стимулирования

Система разработки ПО

- Система разработки программного обеспечения включает в себя *персонал*, *процесс*, *проект* и *продукт*



Процесс и стадии создания ПО

Процесс создания ПО – совокупность мероприятий, целью которых является создание или модернизация ПО

- Анализ предметной области (постановка задачи)
- Разработка проекта системы
 - Создание модели, отражающей основные функциональные требования, предъявляемые к программе
 - Выбор метода решения (построение математической модели)
 - Разработка алгоритма – последовательности действий по решению задачи
- Реализация программы на языке программирования (кодирование)
- Анализ полученных результатов (тестирование)
- Внедрение и сопровождение

Анализ и проектирование

- Этап анализа (analysis) состоит в исследовании системных требований и проблемы

Различают:

- анализ требований (requirements analysis) – исследование требований к системе
- объектный анализ (object analysis) – исследование объектов предметной области
- В процессе проектирования (design) основное внимание уделяется концептуальному решению, обеспечивающему выполнение основных требований
 - Например, на этапе проектирования описываются программные объекты или схема базы данных

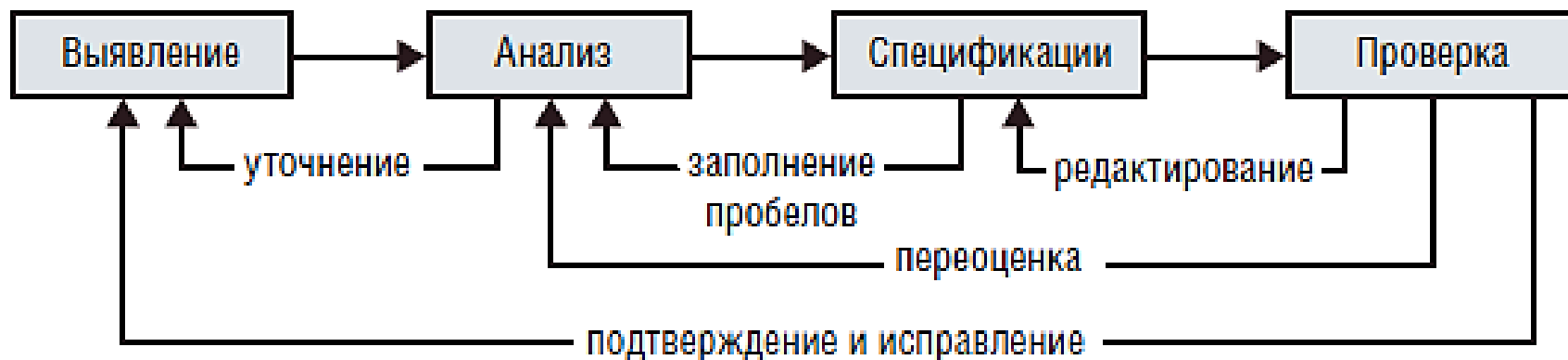
Требования к ПО

IEEE Standard Glossary of Software Engineering Terminology (1990):

- Условия или возможности, необходимые пользователю для решения проблем или достижения целей
- Условия или возможности, которыми должна обладать система или системные компоненты, чтобы выполнить контракт или удовлетворять стандартам, спецификациям или другим формальным документам
- Основная задача этапа определения требований
 - выяснить, обсудить и зафиксировать, что действительно требуется от системы в форме, понятной и клиентам и членам команды разработчиков.

Итеративный процесс формулирования требований

- Разработка требований состоит из выявления, анализа, документирования и проверки



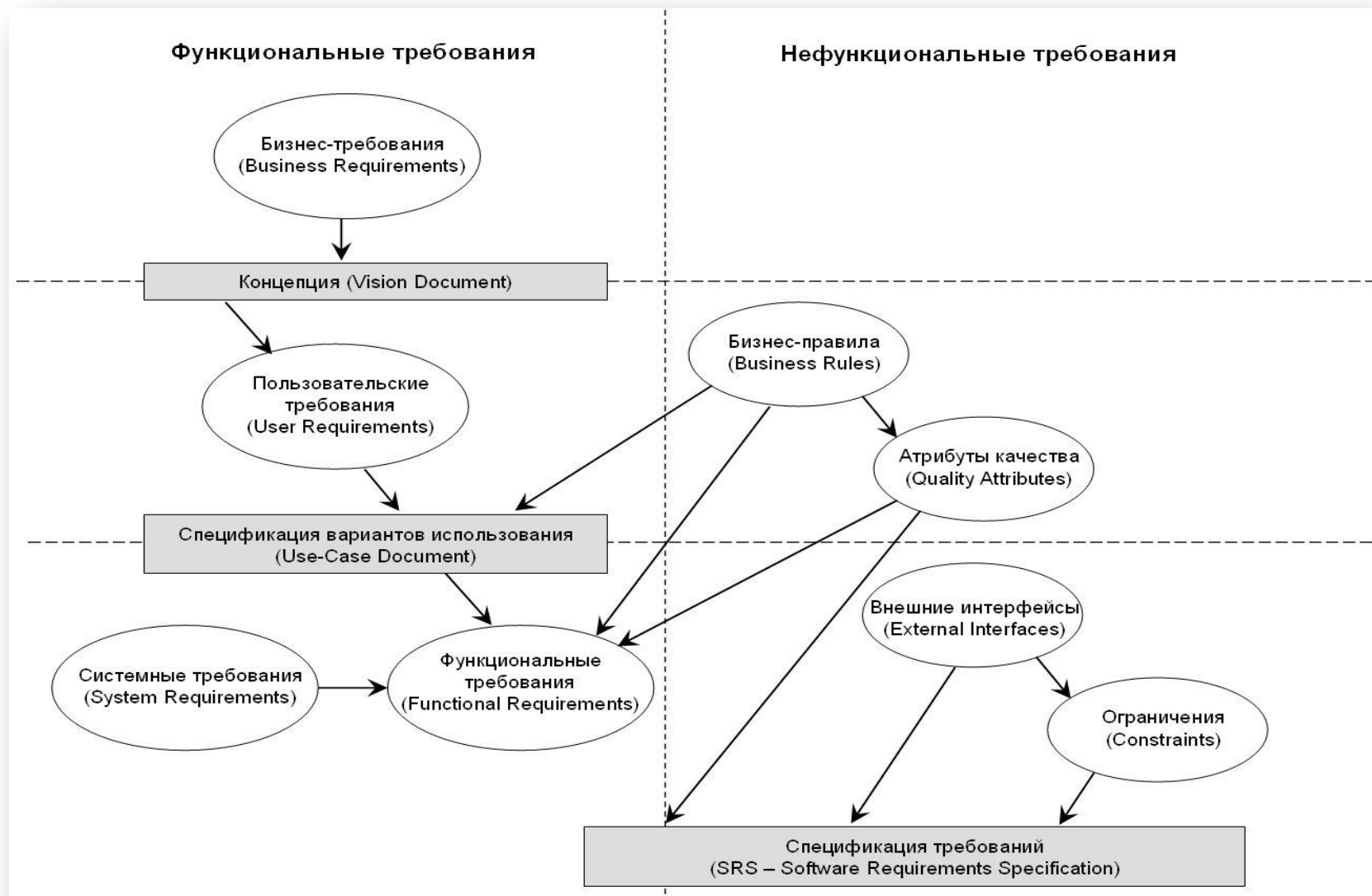
Выявление: задавать клиентам вопросы, слушать их ответы и наблюдать, что они делают


Анализ: классифицировать информацию по категориям, связать потребности клиентов с возможными программными требованиями

Спецификация: структурировать информацию от пользователей и выведенные требования в виде письменных требований-утверждений и диаграмм

Проверка: подтверждение, что «представленное» точно и полно отражает потребности, исправление ошибок

Функциональные и нефункциональные требования к программному средству



- 
- *Бизнес-требования* (business requirements) содержат высокоуровневые цели организации или заказчиков системы
 - *Требования пользователей* (user requirements) описывают цели и задачи, которые пользователям позволит решить система
 - *Функциональные требования* (functional requirements) определяют функциональность ПО, которую разработчики должны построить, чтобы пользователи смогли выполнить свои задачи в рамках бизнес-требований
 - *Системные требования* (system requirements) определяют высокоуровневые требования к продукту, которые содержат многие подсистемы
 - *Бизнес-правила* (business rules) включают корпоративные политики, правительственные постановления, промышленные стандарты и вычислительные алгоритмы
 - *Атрибуты качества* (quality attributes) представляют собой дополнительное описание функций продукта, выраженное через описание его характеристик, важных для пользователей или разработчиков

Пользовательские истории (User story)

- Пользовательские истории составляются в свободной форме, в виде историй или некоторых сценариев использования системы
 - Каждая история имеет условного рассказчика (автора) истории, повествующего о наиболее значимых для исполнения требований к проектируемой программной системе
 - Эти истории являются основой для формулирования функциональных требований
 - Могут быть использованы в дальнейшем для создания приемочных тестов (критериев) при оценивании качества ПО

Пользовательские истории (User story)



Пользовательские истории включают:

- Контекст

- “находясь в окне ... приложения...”

- Событие

- “при выполнении когда делаю.....”

- Результат

- “получаю ответ (уведомление) системы”

Представление требований

- **Документация**, в которой используется четко структурированный и аккуратно используемый естественный язык
- **Графические модели**, иллюстрирующие процессы преобразования, состояния системы и их изменения, отношения данных, логические потоки и т. п.
- **Формальные спецификации**, где требования определены с помощью математически точных, формальных логических языков

Рамки решения и рамки проекта

- Требования к системе («рамки решения») и соответствующие им задачи разработчика («рамки проекта») требования удобно свести в таблицу

URS ID	Описание «рамки решения»	FRS ID	Описание «рамки проекта»
Функция: <i>Название функции</i>			
1	Система обеспечивает возможность пользователям.....	101	Создание меню.....
		102	Реализация просмотра результатов....

Подробные функциональные и нефункциональные требования к продукту записываются в **спецификации к требованиям** к ПО, которая предоставляется тем, кто должен проектировать, разрабатывать и проверять решение.

Как оценить качество требований?

- **Полнота.** Каждое требование должно содержать всю информацию, необходимую разработчику, чтобы реализовать его
- **Корректность.** Каждое требование должно точно описывать возможность, которая будет удовлетворять какую-то потребность заинтересованного лица и четко определять функциональность, которую надо построить
- **Осуществимость.** Необходима возможность реализовать каждое требование при известных возможностях и ограничениях системы и рабочей среды, а также в рамках временных, бюджетных и ресурсных ограничений проекта.

Как оценить качество требований?

- **Необходимость.** Каждое требование должно отражать возможность, которая действительно предоставит заинтересованным лицам ожидаемую бизнес-пользу, выделит продукт на рынке или необходима для соблюдения внешних стандартов, политик или правил.
- **Назначение приоритетов.** Определяйте приоритеты требований на основании важности для получения требуемой пользы.
- **Недвусмысленность.** Читатели должны понимать, о чем идет речь в требовании.
- **Проверяемость.** Требование должно быть проверено объективными методами. Тестировщик должен разработать тесты или применить другие приемы чтобы установить, действительно ли в продукте реализовано каждое требование.

Общий шаблон формулировки требований

- С точки зрения системы:

[необязательное предварительное условие] [необязательный триггер события] **система должна** *[ожидаемая реакция системы]*

- Пример (без триггера).

«Если запрошенный материал есть на складе, система должна отобразить список всех хранимых на складе контейнеров с указанным материалом»

- Можно фразу «система должна» не указывать, если удаление фразы не изменит смысла требования

Общий шаблон формулировки требований

- С точки зрения пользователя:

[класс пользователя или имя действующего лица] **должен иметь возможность** *[выполнить что-то]* *[с каким-то объектом]* *[условия выполнения, время отклика или декларация качества]*

- Пример.

«Менеджер должен иметь возможность повторно заказать любой материал, который он ранее заказывал, путем извлечения и редактирования параметров ранее введенного заказа»

- В требовании рекомендуется использовать название класса пользователя – Менеджер, а не общий термин «пользователь» (явная формулировка максимально снижает вероятность неверного истолкования)

Прецеденты и функциональные требования

- В контексте типов требований основное внимание уделяется функциональным требованиям
- Прецеденты — это в основном функциональные требования, указывающие на то, что должна делать система

Описания прецедентов — это текстовые документы, а не диаграммы.

Моделирование прецедентов — это процесс написания текста, а не рисования

- Однако для иллюстрации имен прецедентов и исполнителей, а также их взаимоотношений в UML определены обозначения для диаграммы прецедентов

Общие рекомендации по формулировке требований

- Избегайте смешения активного и пассивного залогов в попытке сделать материал более интересным для чтения.
- Не обозначайте одно понятие разными терминами, чтобы разнообразить свой текст
- Пишите требования полными предложениями, с правильной грамматикой, правописанием и пунктуацией, предложения и абзацы должны быть краткими и ясными.
- Язык должен быть простым и прямолинейным, характерным для соответствующей предметной области, но не используйте профессиональный жаргон. Определения используемых терминов размещайте в словаре терминов
- Избегайте длинных повествовательных абзацев, которые содержат несколько требований

Стандартизация проектирования ПО

- ГОСТ 34.601-90 - распространяется на автоматизированные системы и устанавливает стадии и этапы их создания
- ISO/IEC 12207 - стандарт на процессы и организацию *жизненного цикла*. Распространяется на все виды заказного ПО
- ГОСТ Р ИСО/МЭК 15288 — 2005 Информационная технология. Системная инженерия. Процессы жизненного цикла систем
- Custom Development Method (методика Oracle)
- Rational Unified Process (RUP)
- Microsoft Solution Framework (MSF)

ISO/IEC 12207

- Международный стандарт ISO/IEC 12207
 - (ISO - International Organization of Standardization - Международная организация по стандартизации,
 - IEC - International Electrotechnical Commission - Международная комиссия по электротехнике).
- ГОСТ Р ИСО/МЭК 2207-99 содержит полный аутентичный текст международного стандарта ISO/IEC 12207

Процессы, определенные в стандарте, образуют множество общего назначения.

Конкретная организация, в зависимости от своих целей, может выбрать соответствующее подмножество процессов для выполнения своих конкретных задач (адаптировать для конкретной организации, проекта или приложения).

Жизненный цикл программного обеспечения

- *Жизненный цикл* - это непрерывный процесс, который начинается
 - с момента принятия решения о необходимости его создания
 - и заканчивается в момент его полного изъятия из эксплуатации.

Структура ЖЦ ПО

Жизненный цикл ПО базируется на трех группах процессов:

■ **основные** процессы

- реализуются под управлением основных сторон (заказчик, поставщик, разработчик, оператор и персонал сопровождения), вовлеченных в жизненный цикл программных средств

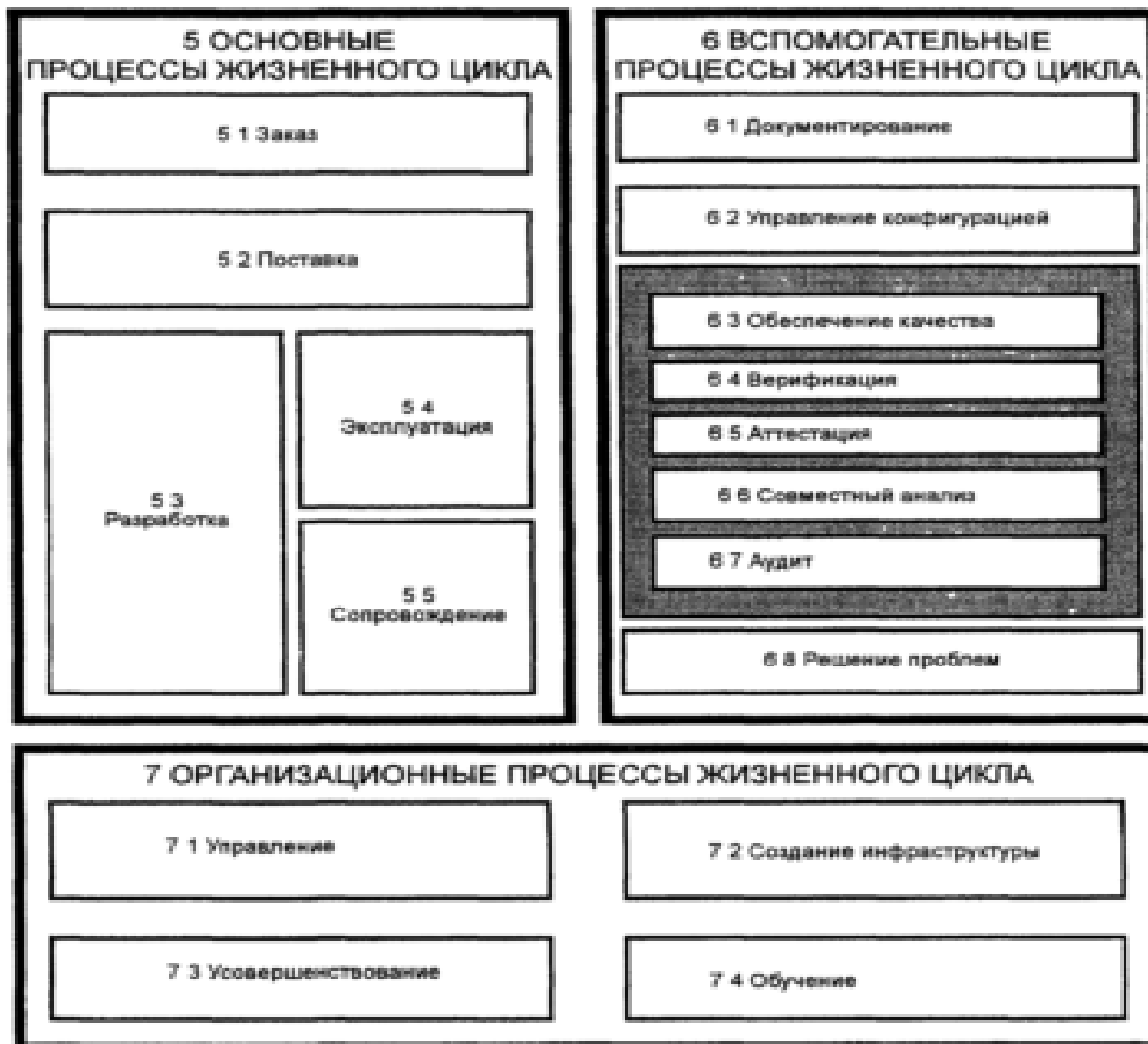
■ **вспомогательные** процессы

- обеспечивают выполнение основных процессов;

■ **организационные** процессы

- применяются для создания, реализации и постоянного совершенствования основной структуры, охватывающей взаимосвязанные процессы жизненного цикла и персонал.

Процессы жизненного цикла



Основные процессы

- Процесс заказа.
 - Определяет работы заказчика.
- Процесс поставки.
 - Определяет работы поставщика.
- Процесс разработки.
 - Определяет работы разработчика.
- Процесс эксплуатации.
 - Определяет работы оператора.
- Процесс сопровождения.
 - Определяет работы персонала сопровождения.
 - Охватывает снятие с эксплуатации программного продукта



Вспомогательные процессы

- Процесс документирования.
- Процесс управления конфигурацией.
- Процесс обеспечения качества
- Процесс верификации.
- Процесс аттестации.
- Процесс совместного анализа.
- Процесс аудита.
- Процесс решения проблемы.



Организационные процессы

- Процесс управления.
- Процесс создания инфраструктуры.
- Процесс усовершенствования.
- Процесс обучения.

Модель жизненного цикла

- ГОСТ Р ИСО/МЭК 15288 — 2005 Информационная технология. Системная инженерия. Процессы жизненного цикла систем
- **Модель жизненного цикла** (life cycle model): Структурная основа процессов и действий, относящихся к жизненному циклу, которая также служит в качестве общей ссылки для установления связей и взаимопонимания сторон.
- **Процесс** (process) – совокупность взаимосвязанных и взаимодействующих видов деятельности, преобразующих входы в выходы



Стадии жизненного цикла

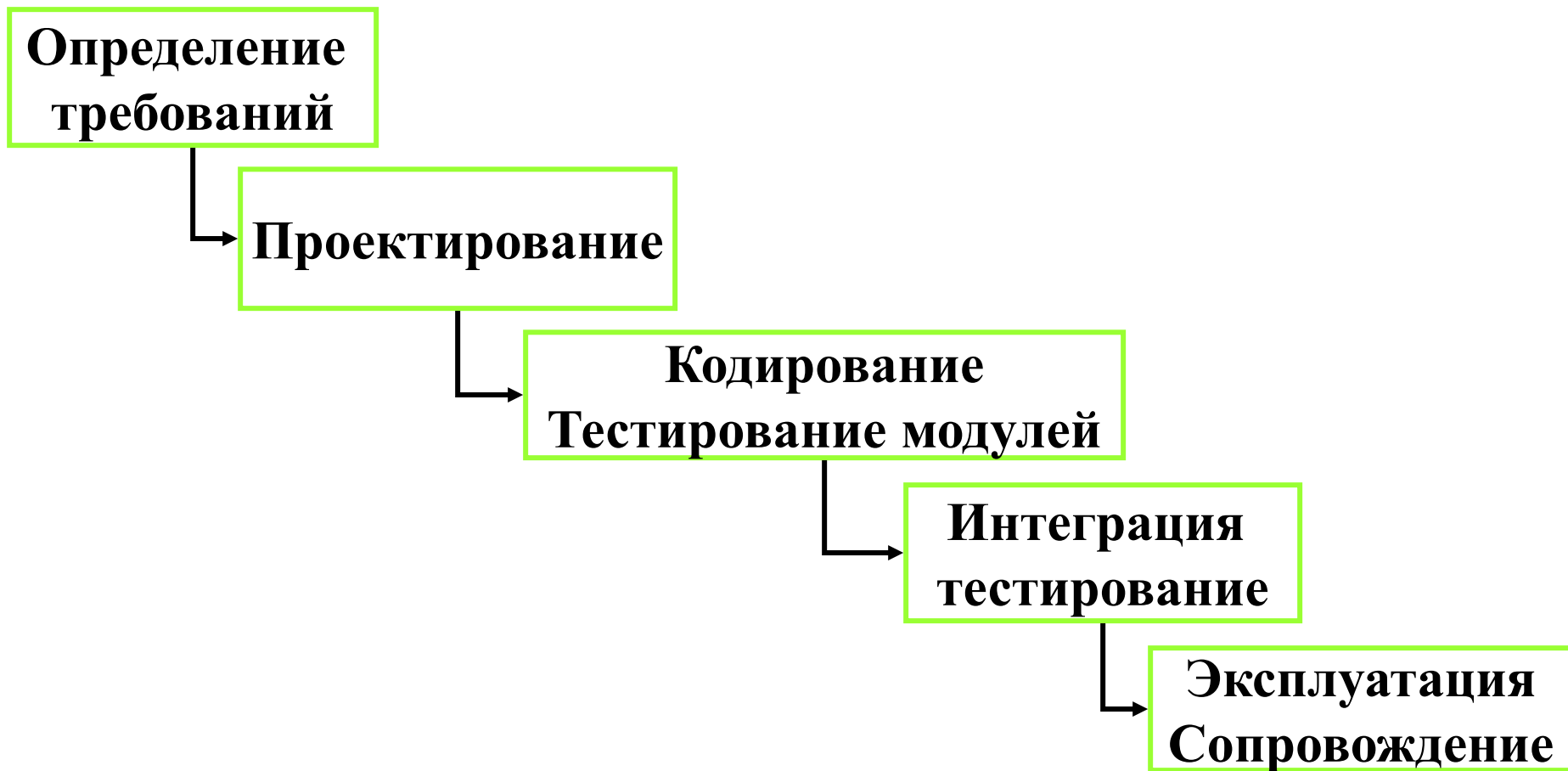
ГОСТ Р ИСО/МЭК 15288 — 2005 (Приложение В)

- Стадия замысла
- Стадия разработки
- Стадия производства
- Стадия применения
- Стадия поддержки применения
- Стадия прекращения применения и списания

Модели процесса

- Классические модели процесса разработки ПО:
 - Каскадная модель (Waterfall model)
фазы выполняются по порядку
 - Эволюционная модель (Evolutionary development)
фазы выполняются по порядку, процесс повторяется

Каскадная модель





■ Каскадная модель:

- ☐ Фиксированный набор стадий
- ☐ Каждая стадия -> законченный результат
- ☐ Стадия начинается, когда закончилась предыдущая.

■ Недостатки: негибкость

- ☐ фаза д.б. закончена, прежде чем приступить к следующей
- ☐ Набор фаз фиксирован
- ☐ Тяжело реагировать на изменения требований

■ Использование: там, где требования хорошо понятны и стабильны.

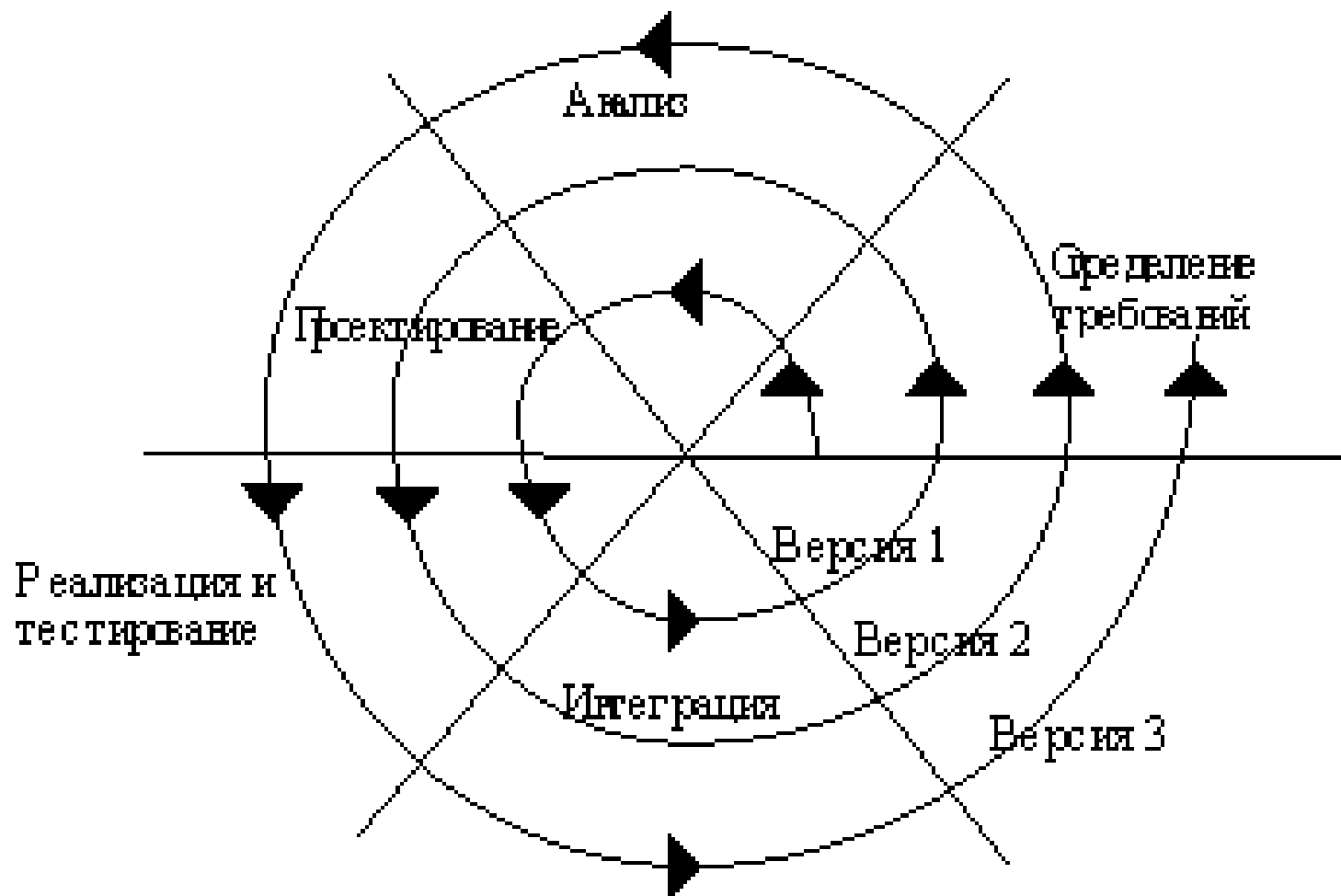
Итерационный подход

- Часто подходы, перечисленные ранее, используется **в совокупности**.
- **Требования** всегда меняются в ходе разработки.
- К каждой из предыдущих моделей можно применить **итерации**.
- Следовательно, важна возможность выполнения итераций, результатом которых является прототип продукта с частичной функциональностью.
- Это достигается в итерационных моделях.
 - Модель пошаговой разработки
 - Спиральная модель разработки

Спиральная модель

- Вместо действий с обратной связью – спираль.
- Каждый виток спирали соответствует 1 итерации.
- Нет заранее фиксированных фаз. В зависимости от потребностей.
- Каждый виток разбит на 4 сектора:
 - Определение целей
 - Оценка и разрешение рисков
 - Разработка и тестирование
 - Планирование
- **Главное отличие: акцент на анализ и преодоление рисков.**
- На каждом витке могут применяться разные модели процесса разработки ПО.

Спиральная модель



Программная документация. Стандарты на разработку прикладных программных средств

- **Стандарт IEEE 830-1998.** Методика составления **спецификаций требований** к программному обеспечению (**SRS**) , рекомендуемая Институтом Инженеров по Электротехнике и Радиоэлектронике (IEEE)
- Методика описывает рекомендуемые принципы составления спецификации требований к программному обеспечению.
- Результат процесса спецификации программного обеспечения является однозначным и полным спецификационным документом.

Стандарт IEEE 830-1998

Основными вопросами, которые должен рассматривать составитель **SRS**, являются следующие:

- **Функциональные возможности.** Каковы предполагаемые функции программного обеспечения?
- **Внешние интерфейсы.** Как программное обеспечение взаимодействуют с пользователями, аппаратными средствами системы, другими аппаратными средствами и другим программным обеспечением?
- **Рабочие характеристики.** Каково быстродействие, доступность, время отклика, время восстановления различных функций программного обеспечения и т.д.?
- **Атрибуты.** Каковы мобильность, правильность, удобство сопровождения, защищенность программного обеспечения?
- **Проектные ограничения,** налагаемые на реализацию изделия. Существуют ли стандарты на эффективном языке реализации, политика по сохранению целостности баз данных, ограничения ресурсов, операционная среда(-ы) и т.д.?

Содержание спецификации требований

■ Указываются

- ☐ функции и возможности, которыми должно обладать ПО
- ☐ необходимые ограничения
- ☐ достаточно подробное описание поведения системы при различных условиях
- ☐ необходимые показатели качества системы, такие как производительность, безопасность и удобство использования.

■ Не должна содержать

- ☐ подробности дизайна, проектирования, тестирования и управления проектом

Спецификация требований служит основой для дальнейшего планирования, дизайна и кодирования, а также базой для тестирования пользовательской документации.

Единая система программной документации

- **Единая система программной документации** - комплекс государственных стандартов, устанавливающих взаимосвязанные правила разработки, оформление и обращения программ и программной документации

Назначение ЕСПД

- В стандартах ЕСПД устанавливают требования, регламентирующие разработку, сопровождение, изготовление и эксплуатацию программ, что обеспечивает возможность:
 - унификации программных изделий для взаимного обмена программами и применение ранее разработанных программ в новых разработках;
 - снижение трудоёмкости и повышение эффективности разработки, сопровождения, изготовления и эксплуатации программных изделий;
 - автоматизации изготовления и хранения программной документации



Состав ЕСПД

В состав ЕСПД входят:

- основополагающие и организационно-методические стандарты
- стандарты, определяющие формы и содержание программных документов, применяемых при обработке данных
- стандарты, обеспечивающие автоматизацию разработки программных документов.

Группы стандартов ЕСПД

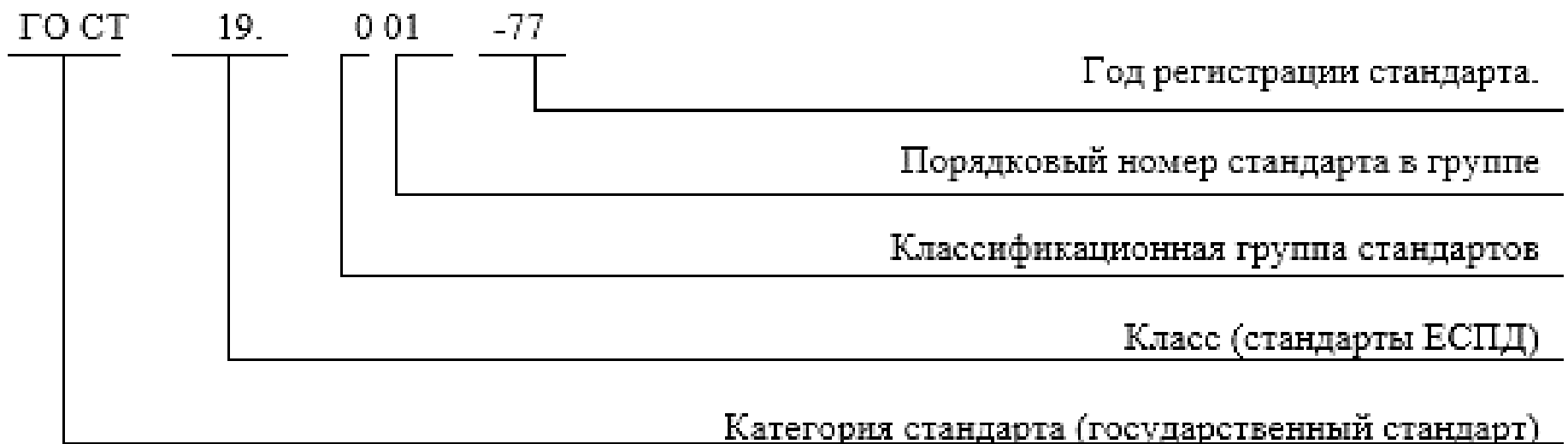
Стандарты ЕСПД подразделяют на группы:

Код группы	Наименование группы
0	Общие положения
1	Основополагающие стандарты
2	Правила выполнения документации разработки
3	Правила выполнения документации изготовления
4	Правила выполнения документации сопровождения
5	Правила выполнения эксплуатационной документации
6	Правила обращения программной документации
7	Резервные группы
8	
9	Прочие стандарты

Обозначение стандарта ЕСПД

Обозначение стандарта ЕСПД должно состоять из:

- цифр 19, присвоенных классу стандартов ЕСПД
- одной цифры (после точки), обозначающей код классификационной группы стандартов
- двузначного числа (после тире), указывающего год регистрации стандарта



Единая система программной документации

- ГОСТ 19.101-87 Виды программ и программных документов
- ГОСТ 19.102-77 Стадии разработки
- ГОСТ 19.201-78. Техническое задание, требования к содержанию и оформлению
- ГОСТ 19.202-78. Спецификация, требования к содержанию и оформлению
- ГОСТ 19.401-78. Текст программы, требования к содержанию и оформлению
- ГОСТ 19.402-78. Описание программы
- ГОСТ 19.501-78. Формуляр, требования к содержанию и оформлению

Программная документация. Стандарты на разработку прикладных программных средств

- ГОСТ 34.601-90. Информационная технология. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Стадии создания.
- ГОСТ 34.602-89 Информационная технология. Комплекс стандартов на автоматизированные системы. Техническое задание на создание автоматизированной системы.
- ГОСТ 34.603-92. Информационная технология. Виды испытаний автоматизированных систем.
- ГОСТ 19. Единая система программной документации.

Техническое задание должно содержать следующие разделы:

- Введение
- Основания для разработки
- Назначение разработки
- Требования к программе или программному изделию
- Требования к программной документации
- Технико-экономические показатели
- Стадии и этапы разработки
- Порядок контроля и приемки

Техническое задание должно содержать следующие разделы:

- Аннотация
- Общие положения
- Общие требования к системе
- Требования к обслуживанию
- Требования к аппаратному обеспечению
- Требования к программному обеспечению
- Требования к обеспечению конфиденциальности и защиты от несанкционированного доступа
- Требования обслуживанию