Щербаков М.В., Щербакова Н.Л.

Основы системной инженерии для бакалавров

Учебное пособие

Диаграммы: <https://app.diagrams.net/#G1ZYyVI1cgdsthhV-L47KHQKEolsC2o0mA>



Волгоград, 2021

УДК 519.816:004.89:004.94

Рецензенты:

кафедра информационных систем и математического моделирования

Волгоградского института управления – филиала Российской академии

народного хозяйства и государственной службы при Президенте РФ,

зав. кафедрой, канд. техн. наук, доцент О. А. Астафурова;

профессор кафедры методики преподавания математики

и физики, ИКТ Волгоградского государственного

социально-педагогического университета,

д-р пед. наук Т. М. Петрова

Печатается по решению редакционно-издательского совета

Волгоградского государственного технического университета

Щербаков, М.В.

Системная инженерия для бакалавров: учеб. пособие / М. В. Щербаков, Н.Л.Щербакова; ВолгГТУ. – Волгоград, 2021. – 64 с.

ISBN 978-5-9948-0000-0

В учебном пособии представлены

Пособие предназначено для студентов бакалавриата обучающихся по направлению информатика и вычислительная техника и развивающие компетенции разработчика сложных информационных систем. Это базовый курс по основам системной инженерии как фундаментального направления знаний для инженера информационных систем.

Ил. 11. Табл. 4. Библиогр.: 46 назв.

ISBN 978-5-9948-0000-0 Волгоградский государственный

технический университет, 2021

М. В. Щербаков, Н.Л. Щербакова 2021

**Оглавление**

[**О курсе: кратко и понятно**](#_y46y4lrlcnol) **6**

[**Лекции: конспекты**](#_51dyx9jmtouo) **8**

[Лекция 1. Введение в системную инженерию.](#_xnnm721d7ux8) 8

[Лекция 2. Жизненный цикл систем.](#_xxmba1bhksja) 9

[Лекция 3. Процесс синтеза систем.](#_4z174wlfox15) 11

[Лекция 4. Целеполагание и ожидания заинтересованных сторон.](#_y1ymi7vyw95t) 12

[Лекция 5. Концептуальная модель.](#_jdk72ws2yuvi) 14

[Лекция 6. Инженерия требований.](#_84fsb1dci4e) 15

[Лекция 7. Архитектура системы.](#_vqes4s1yf33l) 17

[Лекция 8. Верификация и валидация.](#_dfhfspwf6y9o) 18

[**Лабораторный практикум**](#_jw3hlc5v3c8n) **19**

[Лабораторная работа № 1. Введение в системную инженерию.](#_lzgwydcymbor) 19

[ЛР1. Цель работы](#_waodg3vt52o) 19

[ЛР1. Задания на лабораторную работу](#_9s6vk3cuk1l5) 19

[ЛР1. Пояснения к выполнению лабораторной работы](#_pfr62lwfup28) 19

[ЛР1. Контрольные вопросы](#_on2tp4h7q44q) 23

[Лабораторная работа № 2. Жизненный цикл.](#_tclntf2ksawn) 24

[ЛР2. Цель работы](#_u1ddivihwazu) 24

[ЛР2. Задания на лабораторную работу](#_v8iwnei6l23s) 24

[ЛР2. Пояснения к выполнению лабораторной работы](#_v6jxn0tqakxw) 24

[ЛР2. Контрольные вопросы](#_2viq20eahmdx) 28

[Лабораторная работа № 3. Базовые концепции синтеза технических систем](#_ttopx7980m39) 29

[ЛР3. Цель работы](#_7py84fhqto99) 29

[ЛР3. Задания на лабораторную работу](#_sryfb35xmi5f) 29

[ЛР3. Пояснения к выполнению лабораторной работы](#_1fsnry1lwhyg) 29

[ЛР3. Контрольные вопросы](#_p7sxhxelrxxw) 32

[Лабораторная работа № 4. Определение ожиданий заинтересованных сторон и целей создания системы](#_7qpbuekqg054) 33

[ЛР4. Цель работы](#_pe0bxoccgwzf) 33

[ЛР4. Задания на лабораторную работу](#_2hk0w6u8v05i) 33

[ЛР4. Пояснения к выполнению лабораторной работы](#_28jx6f7w4x4n) 33

[ЛР4. Контрольные вопросы](#_pbrptq5erqgw) 37

[Лабораторная работа № 5. Формализация процессов и функций информационных систем](#_9616zclnrhgk) 39

[ЛР5. Цель работы](#_st35ffigdfhr) 39

[ЛР5. Задания на лабораторную работу](#_u9debvnz78lx) 39

[ЛР5. Пояснения к выполнению лабораторной работы](#_aj3yo72nvnx9) 39

[ЛР5. Контрольные вопросы](#_8kt2ww7nhcbz) 42

[Лабораторная работа № 6. Инженерия требований](#_cjq8khxu9k9f) 43

[ЛР6. Цель работы](#_adm9p4825slt) 43

[ЛР6. Задания на лабораторную работу](#_pzo9f4gqdw8b) 43

[ЛР6. Пояснения к выполнению лабораторной работы](#_xtxop7jzer9l) 43

[ЛР6. Контрольные вопросы](#_9sc12og4wx0g) 46

[Лабораторная работа № 7. Концептуальная модель информационной системы](#_pb8h1la1so2k) 47

[ЛР7. Цель работы](#_v32fg6qbvj77) 47

[ЛР7. Задания на лабораторную работу](#_864q3gavh15u) 47

[ЛР7. Пояснения к выполнению лабораторной работы](#_wtm86brtg0ht) 47

[ЛР7. Контрольные вопросы](#_albi0x5e8o86) 50

[Лабораторная работа № 8. Испытания системы](#_vbnhexyjqfsd) 51

[ЛР8. Цель работы](#_i0jaw2t98n6i) 51

[ЛР8. Задания на лабораторную работу](#_uviuieo5kymq) 51

[ЛР8. Пояснения к выполнению лабораторной работы](#_1eycch8ezcdk) 51

[ЛР8. Контрольные вопросы](#_rzva039ayt5) 56

[Оформление результатов выполнения лабораторной работы](#_r8qs1fd4ifyc) 57

[Основные требования](#_vlobax4my3he) 57

[Шаблон оформления протокола лабораторной работы](#_hslh8vodgqw9) 57

[Перечень заданий на выполнение лабораторных работ](#_g3wnwu8ctin) 58

[Темы для выполнения задания](#_2og3vp82305) 58

[**Курсовой проект: методические указания по выполнению**](#_rc6rh0b0h8ua) **61**

[Общая информация](#_icx3zdbtee8q) 61

[Задачи курсового проекта](#_wfn48mc3los8) 61

[Оценка курсового проекта](#_w4nhdt2pumxm) 62

[Требования к разделам](#_jg6srg2e4t4h) 62

[Темы](#_2xqm1pm8yhy) 63

[Перечень тем (названия информационных систем)](#_pmre8pdam7ky) 63

[Порядок выбора темы](#_bnpmsqol8diz) 64

[**Литература**](#_p637k81nrbi7) **66**

[**Выкинул**](#_sqgk5n2vauh8) **67**

[Шаблон выполнения задания Л3.1](#_ql6gkvfpsut) 67

[Пример выполнения](#_ljr1rrl1v9uo) 68

[**Контрольные вопросы (тестирование по курсу)**](#_97mhpnxsf106) **69**

[Learning Objectives](#_35ys3bu4u7qb) 70

[Required Text](#_zg91l3lc1348) 71

[Course Structure](#_uu7yd2nutx53) 71

[Late Policy](#_zghgyy943who) 72

[Grade Components](#_o44e1p43f8qq) 72

[Technical Material (35% of Grade)](#_atwsts7vttpb) 72

[Communication + System Design and Analysis (40% of Grade)](#_lom6hr6qx12x) 73

[Participation (25% of Grade)](#_95y0j2ts5qqn) 73

# 

# О курсе: кратко и понятно

*Зачем этот курс?*

Это базовый курс по основам системной инженерии как фундаментального направления знаний для инженера информационных систем.

*На что могут рассчитывать студенты, прошедшие данный курс?*

Студент поймет основные и инвариантные принципы организации систем, познакомится с классическими подходами к анализу и синтезу технических и информационных систем.

Данные принципы являются базовыми, на которые укладываются подходы программной инженерии и инженерии информационных систем.

*Что дает этот курс?*

Основные понятия о процессах анализа и синтеза систем: от замысла и целеполагания до верификации, валидации и “доставки” системы пользователю.

*Какие компетенции развивает курс системной инженерии?*

* УК-2 Способен определять круг задач в рамках поставленной цели и выбирать оптимальные способы их решения, исходя из действующих правовых норм, имеющихся ресурсов и ограничений
* УК-3 Способен осуществлять социальное взаимодействие и реализовывать свою роль в команде
* ПК-2 Способен анализировать и сопровождать требования к системе, разрабатывать технические задания на создание и модернизацию систем, проектировать системы малого, среднего и крупного масштаба и сложности
* ПК-4 Способен проектировать программное обеспечение и базы данных, разрабатывать и отлаживать программный код, преобразовать программный код и структуры данных для повышения их эффективности, проводить рефакторинг программных приложений и баз данных
* ПК-6 Способен управлять проектами разработки и модернизации систем, выявлять риски, управлять процессами разработки и сопровождения требовании к системам и управлять качеством систем

*Сколько длится курс?*

Курс длится один семестр.

*Как организован курс?*

Курс состоит из 8 лекций и 8 лабораторных работ и задания -- курсового проекта. Реализация курса осуществляется в соответствии с графиком:

* Лекции: 1 лекция в 1 неделю;
* Лабораторные работы: 1 лабораторная работа в 2 недели;
* Курсовой проект: выполняется на протяжении семестра;
* Экзамен (в экзаменационной сессии).

*Какие дисциплины необходимы для освоения системной инженерии?*

Для освоения дисциплины по системной инженерии для бакалавров студент должен освоить дисциплины:

* Б1.В.14 Системный анализ
* Б1.О.01 Архитектура вычислительных систем
* Б1.О.07 Информатика
* Б1.В.08 Моделирование систем
* Б1.В.15 Системы поддержки принятия решений

*Каким образом оценивается выполнение заданий?*

|  |  |  |
| --- | --- | --- |
| Активность | Баллов (максимум) | Итого (максимум) |
| Лабораторные работы | 5 | 40 |
| Курсовой проект | 20 | 20 |
| Экзамен | 40 | 40 |

# 

# Лекции: конспекты

## Лекция 1. Введение в системную инженерию.

*Объекты и системы. Неопределенность и сложность. Анализ систем. Синтез систем. Базовые способы описания систем. Системная схема. Онтология.*

Содержание лекции

1. Что такое системная инженерия и почему она так важно?
2. Объекты и системы.
   1. Понятие объекта, субъекта, системы.
   2. Развитие теории систем (системного анализа).
   3. От теории систем с системной инженерии.
   4. Свойства систем.
3. Неопределенность и сложность.
   1. Неопределенность окружающего мира.
   2. Накопление знаний как путь борьбы с неопределенностью.
   3. Что такое сложность?
   4. Почему сложность -- проблема при разработки любых систем.
4. Анализ систем.
   1. Анализ систем как способ борьбы со сложностью.
   2. Декомпозиция.
   3. Эпистемологические уровни. Почему важно не смешивать.
5. Синтез систем.
   1. Творчество. Творение. Синтез.
   2. Техническое творчество.
   3. Абстракции как инструмент синтеза и борьбы со сложностью.
6. Базовые способы описания систем.
   1. Теоретико-множественное описание.
   2. Математическое моделирование.
   3. Статистические модели.
7. Системная схема.
   1. Понятие системной схемы.
   2. Целевая система.
   3. Использующие системы.
   4. Понятие заинтересованных сторон.
   5. Обеспечивающие системы.
   6. Системы в операционном окружении.
8. Онтология.
   1. Онтологическая модель как способ описания системы.
   2. Основные компоненты онтологической модели.
   3. Примеры построения онтологии.

## **Лекция 2. Жизненный цикл систем**.

Понятие жизненного цикла. Основные термины и определения. Стандарты формального представления жизненного цикла ГОСТ Р 57193-2016 *Системная и программная инженерия. Процессы жизненного цикла систем*. Стоимость владения системы. Базовые модели жизненного цикла разработки систем. Представление и имплементация.

1. Понятие жизненного цикла.
   1. Жизненный цикл биологических систем.
   2. Жизненный цикл технических систем: жизненный цикл (life cycle): Развитие системы, продукции, услуги, проекта или другой создаваемой человеком сущности от замысла до списания.
   3. S-кривая развития технических систем.
   4. Модификация систем.
2. Основные термины и определения:
   1. деятельность, действие (activity)
   2. **базовая линия** (baseline):
   3. концепция функционирования, эксплуатации (concept of operations):
   4. характеристика проекта (design characteristic):
   5. заинтересованная сторона
   6. интерес (системы) (concern)
   7. проектирование
   8. выход, выходной результат процесса (process outcome):
   9. ресурс (resource):
   10. требование (requirement):
   11. риск (risk):
   12. компромисс (trade-off):
3. Стандарты формального представления жизненного цикла ГОСТ Р 57193-2016 *Системная и программная инженерия.* 
   1. ***Модель жизненного цикла системы.*** модель жизненного цикла (life cycle model): Структурная основа процессов и действий, относящихся к жизненному циклу, которая также служит в качестве общего эталона для установления связей и понимания.
   2. *Стадии жизненного цикла системы*
4. *Процессы жизненного цикла систем*.
   1. Понятие процесса.
   2. Формализованное описание процесса.
   3. Группа технических процессов.
   4. Группа процессов технического управления.
   5. Группа процессов соглашение.
5. Стоимость владения системы.
   1. Понятие стоимости владения системы.
   2. Примеры оценки стоимости владения системы.
6. Базовые модели жизненного цикла разработки систем (реализации технических процессов жизненного цикла систем).
   1. Каскадная модель.
   2. Спиральная модель
   3. V-модель.
7. Представление и имплементация.
   1. Понятие описания системы.
   2. Описание архитектуры.
   3. 7 альф системной инженерии.

## **Лекция 3. Процесс синтеза систем**.

История развития практик синтеза сложных систем. Основные этапы процесса синтеза систем. Понятие артефактов. Интеграция. Валидация и верификация. Итерационность разработки и гибкие технологии.

1. История развития практик синтеза сложных систем.
   1. Техническая новизна комбинаторного типа.
   2. Методы анализа и синтеза проектных решений. Морфологическая матрица
   3. Принцип аналогии.
   4. Основы ТРИЗ:
      1. идеальный конечный результат.
      2. техническое противоречие.
      3. технических прием.
2. Основные этапы процесса синтеза систем.
   1. процесс определения целесообразности создания системы.
   2. процесс проектирования.
   3. процесс реализации.
   4. процесс “поставки”.
   5. процесс сопровождения.
3. Понятие артефактов процессов.
   1. Артефакты.
   2. База знаний артефактов.
   3. Подходы к единому информационному хранилищу и описанию систем.
4. Интеграция.
   1. Описание архитектуры системы.
   2. Компоненты и модули системы.
   3. Интеграция компонент.
5. Валидация и верификация.
   1. Процесс валидации.
   2. Процесс верификации.
   3. Представление ожиданий.
6. Итерационность разработки и гибкие технологии.
   1. Agile манифест
   2. Принципы гибкой разработки.
   3. Непрерывное развертывание и непрерывная интеграция.

## **Лекция 4. Целеполагание и ожидания заинте**ресованных сторон.

Human-centric approach.Заинтересованные стороны.Влияние и вовлечение заинтересованных стороны в системном окружении. Интересы и ожидания заинтересованных сторон. Описание высокоуровневых целей. Формализация цели с точки зрения проектного управления. Декомпозиция целей. Управления заинтересованными сторонами как элемент системной инженерии. Инструменты управления заинтересованными сторонами.

1. Human-centric approach.
   1. Рациональный подход к созданию технических систем.
   2. Два мира, два подхода: западный (человеко-центричный) и восточный (общество-центричный) подходы.
   3. В центре разработки системы -- человек.
2. Заинтересованные стороны.
   1. Определение заинтересованных сторон.
   2. Примеры заинтересованных сторон в сложных системах.
   3. Классификация заинтересованных сторон с позиции жизненного цикла систем.
3. Влияние и вовлечение заинтересованных стороны в системном окружении.
   1. Понятие и градация влияния заинтересованных сторон.
   2. Понятие и градация вовлечение заинтересованных сторон.
   3. Матрица “влияние-вовлечение” заинтересованными сторонами.
4. Интересы и ожидания заинтересованных сторон.
   1. Понятие интереса заинтересованных лиц.
   2. Проблемные интервью.
   3. Интерес -- как элемент формального описания архитектуры системы.
   4. Ожидания заинтересованных сторон. Stakeholder Expectations Definition
5. Проблема и барьеры.
   1. Проблемы
   2. Внутренние и внешние барьеры
6. Описание высокоуровневых целей.
   1. Цель как состояние при достижении интереса.
   2. Миссия. Видение. Высокоуровневые цели.
   3. Критерии оценки высокоуровневых целей.
   4. Связь между высокоуровневыми целями и интересом.
7. Формализация цели с точки зрения проектного управления.
   1. Качественные и количественные цели.
   2. Формальное описание целей
   3. Способы оценки достижения цели. Критерии и показатели.
   4. Форсайт
8. Декомпозиция целей.
   1. Декомпозиция высокоуровневых целей.
   2. Иерархия целей.
9. Управления заинтересованными сторонами как элемент системной инженерии.
   1. Процессы управления заинтересованными сторонами
   2. Процесс определения заинтересованных сторон
   3. Процесс планирования управления заинтересованными сторонами
   4. Процесс управление вовлечением заинтересованных сторон
   5. Процесс контроля вовлечения заинтересованных сторон
10. Инструменты управления заинтересованными сторонами.
    1. Канвас: Empathy Map Canvas
    2. Канвас: Value Proposition Canvas

## Лекция 5. **Концептуальная модель**.

Концепция. Понятие концептуальной модели. Принципы создания концептуальной модели. Моделирование. Способы представления концептуальной модели. Функциональные модели. Метамодель.

1. Концепция.
   1. Что такое концептуальное описание системы?
   2. Цель разработки концепции системы (концептуальный дизайн)
   3. Анализ существующих подходов.
   4. Systematic Mapping Study.
2. Понятие концептуальной модели.
   1. Концептуальная модель системы.
   2. Принципы создания концептуальной модели системы как системы обработки информации.
   3. Альтернативные концептуальные модели.
   4. Взгляд на технические системы: И-ИЛИ деревья
   5. Процесс выбора: морфологическая матрица.
3. Способы представления концептуальной модели.
   1. Способы представления системы в виде моделей.
   2. Информационные цепи. Сети Петри как модель представления информационной системы.
   3. Диаграммы потоков данных.
4. Функциональные модели
   1. Схемы функциональной структуры
   2. Сценарии использования
   3. Нотация IDEF0
5. Компонентная диаграмма.
6. Метамодель.
   1. Понятие метамодели
   2. Приложения метамодели
      1. обеспечение единого видения понятий, терминов, областей значений и смыслов значений;
      2. продвижение общего понимания описанных данных;
      3. обеспечение совместного и повторного использования контекстов реализаций.
   3. Метамодель регистра и основные атрибуты.

## **Лекция 6. Инженерия требований**.

Термины, определения и сокращения.Типизация требований. Процесс управления требованиями. Процесс разработки проектных требований. Исходные данные для задания требований. Детализация требований (low-level requirements).

1. Термины, определения и сокращения.
2. Типизация требований.
   1. Технические (функциональные требования) Technical Requirements – Allocation Hierarchically to PBS
      1. Functional Requirements,
      2. Performance Requirements,
      3. Interface Requirements
   2. Операционные требования Operational Requirements – Drive Functional Requirements (
      1. Mission Timeline Sequence
      2. Mission Configurations
      3. Command and Telemetry Strategy
   3. Требования надежности Reliability Requirements – Project Standards – Levied Across Systems
      1. Mission Environments
      2. Robustness, Fault Tolerance, Diverse Redundancy
      3. Verification
      4. Process and Workmanship
   4. Требования безопасности Safety Requirements – Project Standards – Levied Across Systems
      1. Orbital Debris and Reentry
      2. Planetary Protection
      3. Toxic Substances
      4. Pressurized Vessels
      5. Radio Frequency Energy
      6. System Safety
   5. Специальные требования Specialty Requirements – Project Standards – Drive Product Designs
      1. Producibility
      2. Maintainability
3. Процесс управления требованиями.
   1. Процесс разработки требований.
   2. Процесс контроля требований.
4. Процесс разработки проектных требований.
   1. формализованное описание функций объекта,
   2. технические характеристики,
   3. описание принципов и режимов работы, условий эксплуатации,
   4. характеристики взаимодействия с внешними объектами (интерфейсы).
5. Формализация опыта пользователя
   1. User Customer Journey
   2. Пользовательские истории
6. Исходные данные для задания требований.
   1. Основные характеристики требований
   2. Основные правила формулирования текстовых требований
   3. Основные характеристики спецификации требований
   4. Атрибуты требований (метаданные проекта)
7. Детализация требований (low-level requirements).

## **Лекция 7. Архитектура системы**.

Понятие архитектуры. Концептуальная модель описания архитектуры. Основные компоненты архитектуры информационной системы. Связь компонент архитектуры с функциями. Компонентная диаграмма.

1. Понятие архитектуры.
   1. Что такое архитектура?
   2. Что такое процесс архитектуризации.
   3. Основные термины и определения.
2. Немного о потоках данных
   1. Потоки данных
   2. Диаграммы потоков данных (Data Flow Diagram)
3. Концептуальная модель описания архитектуры
   1. Контекст описания архитектуры
   2. Описания архитектуры.
   3. Модели архитектуры
4. Основные компоненты архитектуры информационной системы.
   1. Статические диаграммы.
   2. Динамические диаграммы.
5. Связь компонент архитектуры с функциями системы.
6. Компонентная диаграмма.
   1. Основные элементы компонентной диаграммы
   2. Примеры построения компонентной диаграммы.

## 

## **Лекция 8. Верификация и валидация**.

Интеграция компонент системы (системная интеграция). Процесс верификации.Процесс валидации. Фиксация расхождений в ожидаемых результатах. Оценка качества систем и программного обеспечения. Выводы по курсу.

1. Интеграция компонент системы (системная интеграция).
   1. Понятие компонент и модулей.
   2. Процесс поставки. Управление поставкой.
   3. Процесс интеграции.
2. Процесс верификации.
   1. Понятие верификации.
   2. Верификация в V-модели жизненного цикла.
   3. Основные элементы процесса верификации.
3. Процесс валидации.
   1. Понятие валидации.
   2. Валидация в V-модели жизненного цикла.
   3. Основные элементы процесса валидации.
4. Фиксация расхождений в ожидаемых результатах.
5. Оценка качества систем и программного обеспечения
   1. Эталонные модели оценки качества программного продукта
   2. Эталонная модель процесса оценки качества программного продукт
   3. Стандарт ГОСТ Р ИСО/МЭК 25040-2014 Информационные технологии (ИТ). Системная и программная инженерия. Требования и оценка качества систем и программного обеспечения (SQuaRE). Процесс оценки
   4. Р ИСО/МЭК 25051-2017 Информационные технологии (ИТ). Системная и программная инженерия. Требования и оценка качества систем и программного обеспечения (SQuaRE). Требования к качеству готового к использованию программного продукта (RUSP) и инструкции по тестированию ГОСТ Р от 26 мая 2017 г. № ИСО/МЭК 25051-2017
6. Выводы по курсу.

# Лабораторный практикум

## Лабораторная работа № 1. Введение в системную инженерию.

Время выполнения: 4 часа.

### ЛР1. Цель работы

получить навыки построения основных схем описания информационных систем.

### ЛР1. Задания на лабораторную работу

1. Выбрать информационную систему[[1]](#footnote-0).
2. Построить структурную схему информационной системы.
3. Выполнить расширение структурной схемы с указанием свойств (атрибутов) и основных функций системных элементов.
4. Построить матрицу смежности на основе построения отношений: элемент А -- отношение -- элемент Б.
5. Построить системную схему информационной системы: целевая система - использующая - обеспечивающая - системы в операционном окружении
6. Построение временной диаграммы для учета временного аспекта и изменения системы во времени: представление системной схемы во времени.
7. Привести результаты (что сделано) и выводы (чему научились).

### ЛР1. Пояснения к выполнению лабораторной работы

Системная инженерия (systems engineering): Междисциплинарный подход, управляющий полным техническим и организаторским усилием, требуемым для преобразования ряда потребностей заинтересованных сторон, ожиданий и ограничений в решение и для поддержки этого решения в течение его жизни.

*Система (system): Комбинация взаимодействующих элементов, организованных для достижения одной или нескольких поставленных целей.*

Примечания. Система может рассматриваться как какой-то продукт или как предоставляемые услуги, обеспечивающие этот продукт. На практике, интерпретация данного термина зачастую уточняется с помощью ассоциативного существительного, например система самолета. В некоторых случаях слово система может заменяться контекстно зависимым синонимом, например самолет, хотя это может впоследствии затруднить восприятие системных принципов.

Системный элемент (system element): Представитель совокупности элементов, образующих систему.

*Пример - Системный элемент может представлять собой технические и программные средства, данные, людей, процессы (например, процессы для обеспечения услуг пользователям), процедуры (например, инструкции оператору), средства, материалы и природные объекты (например, вода, живые организмы, минералы) или любые их сочетания.*

Примечание - Системный элемент является отдельной частью системы, которая может быть создана для полного выполнения заданных требований.

С точки зрения системного анализа систему можно представить в виде структурной схемы. Структурная схема является моделью описания системы.

Структурная схема -- представляется графом, вершины (незакрашенные) которого являются компонентами (составными частями) системы, а дуги -- связи между компонентами.

Для структурной схемы привести описание в виде таблицы с колонками:

* mrId (уникальный идентификатор) компонента системы;
* название системного элемента;
* назначение;
* тип информационного процесса (обработка, передача, хранение)

|  |  |  |  |
| --- | --- | --- | --- |
| **mrId** | **Название** | **Назначение** | **Тип информационного процесса** |
|  |  |  |  |

Для структурной схемы привести описание связей в виде таблицы с колонками:

* mrId связи;
* название связи;
* характеристика передаваемой информации.

Связь между системными элементами также будем называть “отношение”. Таким образом для каждой пары вершин (A,B), соединенными отношениями можно сформировать запись вида: A-R(A,B)-B (элемент А -- отношение -- элемент B).

Матрица смежности - квадратная матрица, с числом столбцов и строк равным числу компонент. Элемент матрицы -- отношение. Если отношения нет, то элемент матрицы остается пустым.

|  |  |  |  |
| --- | --- | --- | --- |
|  | A | B | C |
| A | R(A,A) | R(A,B) |  |
| B | R(B,A) |  | R(B,C) |
| C |  | R(C,B) |  |

Атрибуты компонент --- свойства компонент, которые интересны с точки зрения системного инженера.

Главная функция системы -- основная функция, убрав которую из системы, система теряет свое предназначение.

Расширение структурной схемы: в граф системной схемы добавляются:

* вершины (полностью закрашенные) с указанием свойств (атрибутов)
* вершины (заштрихованные) с указанием основный функции компонент системы.

Системная схему -- схема включающая в себя 4 группы систем:

* целевая система[[2]](#footnote-1);
* использующая (системы, которая используют целевую систему);
* обеспечивающая (системы, которые обеспечивают корректное функционирование целевой системы);
* системы в операционном окружении (системы, которые влияют на функционирование целевой системы).

*Целевая (рассматриваемая система) (system-of-interest): Система, жизненный цикл которой рассматривается в рамках анализа или синтеза.*

*Обеспечивающая система (enabling system): Система, которая служит дополнением к рассматриваемой системе на протяжении стадий ее жизненного цикла, но не обязательно вносит непосредственный вклад в ее функционирование.*

Например, когда рассматриваемая система вступает в стадию производства, требуется обеспечивающая производственная система. Каждая обеспечивающая система имеет свой собственный жизненный цикл. Настоящий стандарт может применяться для любой обеспечивающей системы, если она представляется в качестве рассматриваемой системы.

Системы изменяются во времени, поэтому важно понимать как и как быстро изменяются системы. Временная диаграмма позволяет представить системы (целевую, использующую, обеспечивающие и системы в операционном окружении) с учетом временного аспекта (представление системной схемы во времени).

|  |  |  |  |
| --- | --- | --- | --- |
|  | прошлое | настоящее | будущее |
| целевая |  |  |  |
| использующая |  |  |  |
| обеспечивающая |  |  |  |
| системы в операционном окружении |  |  |  |

### **ЛР1.** Контрольные вопросы

1. Что такое система? Дайте определение системы.
2. Дайте определение структурной схемы .
3. В чем специфика главной функции системы
4. Приведите пример отношений между элементами системы.
5. Приведите пример системной схемы.
6. Приведите пример использующей системы для целевой системы “программа калькулятор”.
7. Приведите пример обеспечивающей системы для целевой системы “персональный компьютер”.
8. Приведите пример системы в операционном окружении для целевой системы “компьютерная мышь”.
9. Дайте собственное определение временной диаграммы.

## 

## 

## Лабораторная работа № 2. Жизненный цикл.

Время выполнения: 4 часа.

### ЛР2. Цель работы

получить навыки формального описания жизненного цикла системы (студент должен научится и приобрести навык описания процессов и состояний жизненного цикла систем

### ЛР2. Задания на лабораторную работу

1. Выбрать информационную систему[[3]](#footnote-2).
2. Представить систему (на этапе функционирования) в виде множества состояний и переходов между состояниями.
   1. построить схему состояний системы и переходов между ними в виде графа
   2. выполнить описание состояний системы в виде JSON
3. Построить матрицу переходов и оценить вероятности переходов
4. Выделить основные этапы жизненного цикла системы и привести для каждого этапа состояния системы.
5. Выполнить описание состояний системы в жизненном цикле в виде JSON
6. Привести результаты (что сделано) и выводы (чему научились).

### ЛР2. Пояснения к выполнению лабораторной работы

Жизненный цикл (life cycle): Развитие системы, продукции, услуги, проекта или другой создаваемой человеком сущности от замысла до списания.

Модель жизненного цикла (life cycle model): Структурная основа процессов и действий, относящихся к жизненному циклу, которая также служит в качестве общего эталона для установления связей и понимания.

Систему (на этапе функционирования[[4]](#footnote-3)) можно описать различными способами. Например, теоретико-множественное представление системы S может иметь вид S = <{E}, {S}, {P}>, где {E} -- множество элементов, {S} -- множество состояний, {P} -- множество переходов между состоями.

Будем считать, что система в текущий момент времени может находится только в одном из возможных состояний. Состояния и переходы между ними можно отобразить в виде направленного графа, где вершина -- состояние системы, дуги -- переходы между состояниями. Дуги -- взвешенные, вес дуг -- вероятность перехода из одного состояния в другое. Например, p(S0|S1) -- вероятность перехода в состояние S0 из состояния S1.

О диаграммах состояний:

* http://www.agilemodeling.com/artifacts/stateMachineDiagram.htm
* <http://agilemodeling.com/style/stateChartDiagram.htm>
* <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-state-machine-diagram/>

Создать диаграмму можно в

* Astah https://astah.net/,
* Umbrello https://umbrello.kde.org/,
* https://www.draw.io/.

Формализованное описание состояний и переходов между состояниями в формате JSON[[5]](#footnote-4). Обратим внимание на поле $ref -- уникальный идентификатор сущностей (состояний, переходов).

{

"task\_description": "Состояния оборудования",

"system\_drill": {

"name": {

"title": "Сверлильный станок",

"$ref": "system/drill/"

},

"states": {

"state\_up": {

"name": "Работоспособное состояние",

"$ref": "system/drill/state/upoperation/",

"functions": {

"function\_drill": {

"name": "Сверление",

"$ref": "system/drill/state/upoperation/drill"

}

}

},

"state\_failure": {

"name": "Отказ",

"$ref": "system/drill/state/failure/"

}

},

"transitions": {

"transition\_up\_failure": {

"state\_source": "system/drill/state/upoperation/",

"state\_destinations": {

"destination\_01": "system/drill/state/failure/",

"probability": 0.35

}

},

"transition\_failure\_up": {

"state\_source": "system/drill/state/failure/",

"state\_destinations": {

"destination\_01": "system/drill/state/upoperation/",

"probability": 0.5

} } } }}

Для формализации переходов целесообразно построить матрицу смежности и указать значения вероятностей.

|  |  |  |  |
| --- | --- | --- | --- |
| Текущее/Прошлое | S0 | S1 | S2 |
| S0 | p(S0,S0) | p(S0,S1) | p(S0,S2) |
| S1 | p(S1,S0) | p(S1,S1) | p(S1,S2) |
| S2 |  | p(S2,S1) |  |

Этапы жизненного цикла системы зависят от специфики системы, но, как правило, это этапы:

1. замысел;
2. проектирование;
3. реализация;
4. функционирование;
5. модернизация/списание.

Для каждого этапа жизненного цикла целесообразно выделить состояния системы.

|  |  |
| --- | --- |
| **Этап жизненного цикла** | **Состояния** |
|  |  |

Введем на каждом этапе жизненного цикла фиктивное состояние, которое будем называть “вехой”. Из этого состояния можно перейти *только* в состояние следующего этапа жизненного цикла.

Описание состояний системы в жизненном цикле осуществляется строго по приведенному ниже формату JSON.

{

"task\_description": "Этапы жизненного цикла",

"first\_stage": {

"name": {

"title": "Замысел",

"$ref": "stage/stage\_1"

},

"states": {

"state01": {

"name": "Возможные Идеи",

"$ref": "stage/stage\_1/state\_01"

},

"state02": {

"name": "Концепции",

"$ref": "stage/stage\_1/state\_02"

}

},

"transitions": {

"transition\_state01\_state02": {

"state\_source": "stage/stage\_1/state\_01",

"state\_destinations": {

"destination\_01": "stage/stage\_1/state\_02",

"probability": 1

}

},

"transition\_state02\_state02": {

"state\_source": "stage/stage\_1/state\_02",

"state\_destinations": {

"destination\_01": "stage/stage\_1/state\_02",

"probability": 0.5

} } } }}

### ЛР2. Контрольные вопросы

1. Дайте определение понятию жизненного цикла.
2. Дайте определение понятию модели жизненного цикла.
3. Приведите пример состояний системы “поисковик[[6]](#footnote-5)”
4. Приведите пример состояний системы и переход между ними (для произвольной системы)
5. Что такое веха?
6. Назовите основные этапы жизненного цикла информационной системы.
7. Назовите основные этапы жизненного цикла нетехнической системы.

## 

## Лабораторная работа № 3. Базовые концепции синтеза технических систем

Время выполнения: 4 часа.

### ЛР3. Цель работы

получить навык анализа и комбинаторного синтеза простых систем

### ЛР3. Задания на лабораторную работу

1. Модификация структурной схемы системы.
2. Для каждого элемента (подсистемы) выделить 3 альтернативы ее реализации
3. Выполнить расширение структурную схему технической системы до И-ИЛИ графа.
4. Выполнить процедуру синтеза 5 новых технических решения на И-ИЛИ графа.
5. Выделить критерии качества оценки технических решений.
6. Проанализировать 5 новых технических решений: выделить преимущества, недостатки и возможные пути преодоления недостатков.
7. Привести результаты (что сделано) и выводы (чему научились).

### ЛР3. Пояснения к выполнению лабораторной работы

Структурная схема -- представляется графом, вершины (незакрашенные) которого являются компонентами (составными частями) системы, а дуги -- связи между компонентами.

Построенная в первой лабораторной работе структурная схема должна быть проанализиована и модифицирована в соответствии с замечаниями и рекомендациями преподавателя[[7]](#footnote-6). На графе структурной схемы обозначить системные элементы {E} = {E1,E2, …, En}

Для каждого системного элемента предложить 3 альтернативные реализации системного элемента.

Формально необходимо представить в виде кортежей: E1 = {A1\_(E1), A2\_(E1), A3\_(E1)} и в формате таблицы.

|  |  |
| --- | --- |
| **Структурный элемент** | **Альтернативные реализации** |
|  |  |

Альтернативы следует представить в виде кортежа “ключ:значение”. Пример: A1\_(E1) = <k1:v1, k2:v2, k3:v3, ...>.

На основе выделенных для каждого структурного элемента альтернатив выполнить расширение структурной схемы до И-ИЛИ графа.

И-ИЛИ граф -- граф, состоящий из

* И-вершин (незакрашенных), которые характеризуют структурные элементами (составными частями) системы;
* дуги типа “И” -- связи между незакрашенными вершинами (между структурными элементами),
* ИЛИ-вершин (закрашенных), которые характеризуют альтернативы реализации структурных элементов.
* дуги типа “ИЛИ” -- связи между незакрашенными вершинами и соответствующими им закрашенным вершинам (между структурными элементами и альтернативами их реализации).

На основе построенного И-ИЛИ графа Выполнить процедуру синтеза 5 новых технических решений. Формально новое техническое решение необходимо представить в виде кортежей:

Sol\_1 = <E1:A1, E2:A1, E3:A2, E4:A3>,

т.е. для структурного элемента 1 выбрана альтернатива 1, для структурного элемента 2 выбрана альтернатива 1, для структурного элемента 3 выбрана альтернатива 2, для структурного элемента 4 выбрана альтернатива 3.

Определить не менее 2-х критериев оценки технических решений.

*Критерий существенный, отличительный признак, на основании которого производится оценка, определение или классификация чего-л.*

Описание критериев привести в формате таблицы.

|  |  |  |
| --- | --- | --- |
| **Обозначение критерия** | **Название критерия** | **Способ определения критерия** |
| C1 |  |  |
| C2 |  |  |

Проанализировать 5 новых технических решений: выделить преимущества, недостатки и возможные пути преодоления недостатков. Результаты представить в таблице.

|  |  |  |  |
| --- | --- | --- | --- |
| **Решение** | **Преимущества** | **Недостатки** | **Способы преодоления недостатков** |
|  |  |  |  |

Осуществить ранжирование технических решений по каждому из выделенных критериев. Результаты ранжирования представить в виде таблицы.

|  |  |  |
| --- | --- | --- |
|  | **Критерий C1** | **Критерий C2** |
| Ранг 1 | Sol\_4 | Sol\_4 |
| Ранг 2 | Sol\_1 | Sol\_5 |
| Ранг 3 | Sol\_2 | Sol\_1 |
| Ранг 4 | Sol\_3 | Sol\_2 |
| Ранг 5 | Sol\_5 | Sol\_3 |

### ЛР3. Контрольные вопросы

1. Дайте определение альтернативы.
2. Дайте определение И-ИЛИ графа.
3. Приведите пример И-ИЛИ графа.
4. Опишите процедуру синтеза новых технических решения на И-ИЛИ графе.
5. Дайте определение критерии качества оценки технических решений.
6. Приведите пример критерии качества оценки технических решений.

## Лабораторная работа № 4. Определение ожиданий заинтересованных сторон и целей создания системы

Время выполнения: 4 часа.

### ЛР4. Цель работы

приобрести навык формализации предпочтений заинтересованных сторон системы

### ЛР4. Задания на лабораторную работу

1. Определить перечень заинтересованных сторон в функционировании рассматриваемой системы и определить их интерес.
2. Сформулировать цели создания системы.
3. Разработать пользовательские истории (сценарии использования) для целевой системы (не менее 5 пользовательские истории для каждого интереса).
4. Разработать диаграмму использования (UML use case diagram)
5. Привести результаты (что сделано) и выводы (чему научились).

### ЛР4. Пояснения к выполнению лабораторной работы

Заинтересованная сторона, правообладатель (stakeholder): Индивидуум или организация, имеющие право, долю, требование или интерес в системе или в обладании ее характеристиками, удовлетворяющими их потребности и ожидания.

Пример - Конечные пользователи, организации конечного пользователя, поддерживающие стороны, разработчики, производители, обучающие стороны, сопровождающие и утилизирующие организации, приобретающие стороны, организации поставщика, органы регуляторов.

Примечание - Некоторые заинтересованные стороны могут иметь противоположные интересы в системе.

Заказчик (customer): Организация или лицо, получающие продукт или услугу.

Приобретающая сторона (acquirer): Заинтересованная сторона, которая приобретает или получает продукт или услугу от поставщика.

Интерес (системы) (concern): Польза или проблемы в системе, относящиеся к одной или нескольким заинтересованным сторонам.

Примечание - Интерес относится к любому воздействию на систему в ее окружающей среде, включая воздействия разработки, технологические, деловые, эксплуатационные, организационные, политические, экономические, юридические, регулирующие, экологические и социальные воздействия.

Управление заинтересованными сторонами проекта включает в себя процессы, необходимые для выявления людей, групп и организаций, которые могут оказывать или на которых может оказывать воздействие проект, для анализа ожиданий заинтересованных сторон и их воздействия на проект, а также для разработки соответствующих стратегий управления для эффективного вовлечения заинтересованных сторон в принятие решений и исполнение проекта.

Управление заинтересованными сторонами также сосредотачивается на постоянной коммуникации с заинтересованными сторонами с целью понимания их потребностей и ожиданий, на реагировании на проблемы по мере их возникновения, на управлении конфликтующими интересами и на способствовании соответствующему вовлечению заинтересованных сторон в принятие решений и в операции проекта. Удовлетворенностью заинтересованных сторон следует управлять как одной из ключевых целей проекта.

Компромисс (trade-off): Действия по принятию решений, в ходе которых на основе конечной выгоды правообладателей производится выбор из различных требований и альтернативных решений.

Для целевой системы необходимо выделить заинтересованных сторон и определить интересы для них. Результаты представить в виде таблицы.

|  |  |  |
| --- | --- | --- |
| **Заинтересованная сторона** | **Обозначение интереса** | **Интерес** |
| Диспетчер (Д) | Д.И1 | Отображение информации о текущем состоянии оборудовании |

Необходимо исходить из логики, что назначение системы определяется исходя из интереса заинтересованной стороны.

На основе интересов заинтересованных сторон необходимо сформулировать цели создания системы. Цели могут быть качественные и количественные.

В соответствии с ГОСТ 34.602 в разделе "Цели создания системы" технического задания на создание автоматизированной системы (АС) приводят наименования и требуемые значения технических, технологических, производственно-экономических или других показателей объекта автоматизации, которые должны быть достигнуты в результате создания АС, и указывают критерии оценки достижения целей создания системы.

Пример качественных целей создания информационных систем.

* поддержка основных бизнес-процессов … ;
* организация единого информационного пространства … ;
* повышение эффективности процессов … ;
* снижение времени информационного обмена….

Пример количественных целей:

* снижение времени на …. в 2 раза;
* снижение числа ошибок в 1,7 раза.

Цели изложить в таблице.

|  |  |  |
| --- | --- | --- |
| **Обозначение интереса** | **Формулировка цели** | **Тип цели[[8]](#footnote-7)** |
|  |  |  |

Пользовательские истории (user story) -- формализованное представление интереса заинтересованной стононы в целевой системы, определенное через действие с системой.

Пользовательская история представляются в соответствии с концепцией user story по следующему шаблону:

*Как [пользователь, заказчик, сотрудник] я хочу [информация о действии пользователя с системой] чтобы [формирование цели, ценности, результата действия].*

Ниже приведены примеры пользовательских историй[[9]](#footnote-8).

1. Как главный инженер, я хочу иметь информацию о числе подключенного оборудования к Системе мониторинга, чтобы оценить риски для узлов, неохваченных системой.

2. Как главный инженер, я хочу иметь информацию о подключенном оборудовании к Системе мониторинга ГТУ для оценки масштаба внедрения Системы.

3. Как главный инженер, я хочу иметь перечень измеряемых параметров для каждого оборудования, для оценки полноты сбора данных о текущем состоянии оборудовании.

Для описание пользовательских историй необходимо использовать шаблон в формате JSON

{

"task\_description": "User Stories",

"System Description": {

"title": "System name",

"$ref": "URI #/definitions/name",

"UStories": {

"UStory\_1": {

"Название": "",

"$ref": " User story URI http://mydomain.com/geo.json",

"UStory\_Description": {

"Как": "",

"я хочу": "",

"чтобы": ""

} } } }}

Еще один способ формализации действий заинтересованной стороны с системой -- диаграмма вариантов использования или диаграмма прецедентов (use case diagram) языка UML[[10]](#footnote-9).

На диаграмме вариантов использования изображаются:

* акторы — заинтересованные стороны, взаимодействующие с целевой системой;
* варианты использования (прецеденты) - возможности[[11]](#footnote-10), предоставляющие системой акторам;
* отношения между элементами диаграммы
* комментарии;

.

В самом общем случае, диаграмма вариантов использования представляет собой граф специального вида, который является графической нотацией для представления конкретных вариантов использования, актеров и отношений между этими элементами. При этом отдельные элементы диаграммы заключают в прямоугольник, который обозначает границы проектируемой системы. В то же время отношения, которые могут быть изображены на данном графе, представляют собой только фиксированные типы взаимосвязей между актерами и вариантами использования, которые в совокупности описывают сервисы или функциональные требования к моделируемой системе[[12]](#footnote-11).

Разработать диаграмму вариантов использования для вашей целевой системы с использованием редактора <https://app.diagrams.net/> или любого другого редактора.

### ЛР4. Контрольные вопросы

1. Опишите процесс определения ожиданий заинтересованных сторон.
2. Что такое пользовательская история?
3. Приведите пример пользовательской истории.
4. Приведите пример ролей пользователя.
5. Возможны ли конфликтующие пользовательские истории? Какие способы разрешения подобных конфликтов могут быть применены?
6. Приведите примеры качественных и количественных целей.
7. Приведите пример диаграммы вариантов использования для произвольной системы.

## 

## Лабораторная работа № 5. Формализация процессов и функций информационных систем

Время выполнения: 4 часа.

### ЛР5. Цель работы

приобрести навык формализации процессов и функций систем

### ЛР5. Задания на лабораторную работу

1. Построить формальное описание процесса в формате JSON
2. На основе пользовательских сценариев сформировать функциональные диаграммы по методологии IDEF0 (два уровня):
   1. уровень 1: построить диаграммe A-0 общая (главная) функция;
   2. уровень 2: декомпозиция главной функции в дочерней диаграмме.
3. Привести результаты (что сделано) и выводы (чему научились).

### ЛР5. Пояснения к выполнению лабораторной работы

Процесс (process): Совокупность взаимосвязанных или взаимодействующих видов деятельности, преобразующая входы в выходы.

Цель процесса (process purpose): Устремления высокого уровня в выполнении процесса и вероятные выходы эффективной реализации процесса.

Примечание - Необходимо, чтобы реализация процесса обеспечивала ощутимую пользу заинтересованным сторонам.

Выход, выходной результат процесса (process outcome): Наблюдаемый результат успешного достижения цели процесса.

Формально процесс Pr = <{I}, {O}, {S}> может быть описан как совокупность множества входов {I}, выходов {O} и параметров процесса {S}.

Для рассматриваемой системы описать процессы жизненного цикла в соответствии с шаблоном.

01\_02": {

"name": "Название выхода 2 в операции 1",

"$ref": "Уникальный URI входа 1"

}

}

},

"operation\_02": {

"name": "Название операции 2",

"$ref": "Уникальный URI операции 2",

"inputs\_operation\_02": {

"input\_operation\_02\_01": {

"name": "Название входа 1 в операции 2",

"$ref": "Уникальный URI входа 1"

},

"input\_operation\_02\_02": {

"name": "Название входа 2 в операции 2",

"$ref": "Уникальный URI входа 1"

}

},

"outputs\_operation\_02": {

"output\_operation\_02\_01": {

"name": "Название выхода 1 в операции 2",

"$ref": "Уникальный URI выхода 1"

},

"output\_operation\_02\_02": {

"name": "Название выхода 2 в операции 2",

"$ref": "Уникальный URI выхода 2"

} } } } }}

Ранее для системы была выделена главная функция, а каждого системного элемента были выделены основные функции.

Для формализованного представления функций используется Методология функционального моделирования (IDEF0[[13]](#footnote-12)).

Методология описана в Федеральном стандарте обработки информации[[14]](#footnote-13). Настоящий стандарт описывает язык моделирования IDEF0 (семантика и синтаксис), а также связанные с ним правила и методы разработки структурированных графических представлений системы или предприятия. Использование стандарта позволяет создавать модели, описывающие системные функции (действия, поведение, процессы, операции), функциональные связи и данные (информация или объекты), которые поддерживают интеграцию систем.

В ходе реализации программы ICAM был разработан ряд методов, известных как методы IDEF (iCAM Definition), которые включали в себя следующее:

* Метод IDEF0, используется для создания „функциональной модели“. Функциональная модель это структурированное представление функций, действий или процессов в моделируемой системе или объекте.
* Метод IDEF1, используется для создания „информационной модели“. Информационная модель представляет собой структуру и семантику информации внутри моделируемой системы или объекта.
* Метод IDEF2, используется для создания „динамической модели“. Динамическая модель представляет собой изменяющиеся во времени поведенческие характеристики моделируемой системы или объекта.

Диаграмма A-0: специальный вид однокомпонентной (контекстной) диаграммы IDEF0, состоящей из одного блока, описывающего функцию верхнего уровня, ее вводы, выводы, контроля, и механизмы, вместе с формулировками цели модели и точки зрения, с которой строится модель.

Диаграмма включает в себя функциональный блок (ICOM-блок). Блоки представляют функции, обозначающие что должно быть выполнено. Имя функции должно быть активным глаголом или отглагольным существительным.

Стрелки в блоке нотации IDEF0 могут быть нескольких типов:

* Input (Ввод) (слева)
* Output ( Вывод) (справа)
* Control (Управление) (сверху)
* Mechanism (Механизм) (снизу)

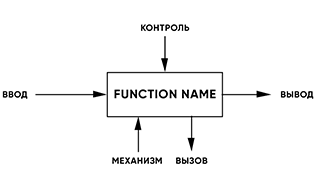


Рисунок. Функциональный блок и стрелки данных / объектов

Для главной функции необходимо построить Диаграмму A-0.

Единственная функция, представленная на контекстной диаграмме верхнего уровня, может быть разложена на основные подфункции путем создания дочерней диаграммы.

Для функций элементов системы построить дочернюю диаграмму.

### ЛР5. Контрольные вопросы

1. Дать определение процесса.
2. Дать определение цели процесса.
3. Привести пример формального представления процесса.
4. Привести пример процесса и декомпозиции на подпроцессы.
5. В чем суть методологии функционального моделирования.
6. Что такое ICOM-блок.
7. Привести пример описания процесса в IDEF0 формализации.
8. Что такое Диаграмма A-0?
9. Что такое дочерняя диаграмма.

## 

## Лабораторная работа № 6. Инженерия требований

Время выполнения: 4 часа.

### ЛР6. Цель работы

приобрести навык формирования функциональный и нефункциональных требований к системе

### ЛР6. Задания на лабораторную работу

1. Сформировать пользовательские сценарии для которых указать базовые действия, альтернативные действия и ошибки;
2. Сформировать функциональные требования;
3. Сформировать нефункциональные требования
4. Привести результаты (что сделано) и выводы (чему научились).

### ЛР6. Пояснения к выполнению лабораторной работы

Пользовательские сценарии -- развитие пользовательских историй для формальное представления функций системы[[15]](#footnote-14).

Общая схема описания пользовательского сценария. Для каждого пользовательского сценария должна быть сформулировано общее описание согласно таблице

|  |  |
| --- | --- |
| Идентификатор (ID) пользовательского сценария |  |
| Название сценария: |  |
| Пользователь/Актер |  |
| Предварительные условия | Условия, выполнение которых запускает выполнение действий. |

Базовый сценарий (базовый сценарий). Оптимальное или нормальное течение событий в рамках пользовательского сценария. Базовый сценарий содержит последовательность действий и событий, приводящих к ожидаемому результату. Базовый сценарий не должен содержать условий и развилок.

|  |  |  |
| --- | --- | --- |
| Номер шага | Действие пользователя | Действие системы |
|  | Описание действия пользователя | Описание отклика системы |

Альтернативные сценарии (может быть более одного) приводящие к такому же ожидаемому результату.

|  |  |  |
| --- | --- | --- |
| Номер шага | Действие пользователя | Действие системы |
|  | Описание действия пользователя | Описание отклика системы |

Исключения. Определяются системные ошибки и ошибки в данных, которые могут возникать на каждом шаге в базовом и альтернативных сценариях.

|  |  |  |
| --- | --- | --- |
| Номер шага | Условия возникновения ошибки | Действие системы |
|  | Описание ситуации при которой возникает системная ошибка или ошибка в данных.  Указывается предшествующий исключению шаг сценария (базового или альтернативного), и последующие шаги сценария (базового или альтернативного), выполняющиеся после возникновения исключения.    <Example: The VIN is invalid. 11(Basic Flow, Step 2)> | Описание действий системы для перехода на следующий шаг или состояние в последовательности действий, а при условие прекращения определение финального состояния системы.  Например. (i.e. redisplay first screen before error condition or something else.) (Number each distinct step). > |

Пост - условия. Определяются условия и действия, требующие выполнения после окончания сценария.

Функциональные требования. Каждый функционал (функциональная возможность (feature) описывается в соответствии со следующей структурой.

<Функциональные требования или функциональная возможность #1>

1. Введение (характеристика функции)
2. Входы
3. Обработка (алгоритм обработки)
4. Выходы
5. Обработка ошибок
6. Тесты (unit тесты, функциональные тесты)

Привести не менее 10 функциональных требований к системе.

Пример

<Сортировка массива целых чисел>

3.2.1.1 Функционал сортировки массива целых чисел

3.2.1.2 Входы: массив из целых чисел произвольной длины

3.2.1.3 Обработка: любым алгоритмом сортировки сложностью O(n log n)

3.2.1.4 Выходы: отсортированный в процессе убывания значений

3.2.1.5 Обработка ошибки: если массив содержит значения None, или значения отличаются от целых (например строки или числа с плавающей запятой), то должно выдаваться сообщение об ошибке.

Нефункциональные требования. Нефункциональные требования могут быть следующих типов:

* производительность;
* надежность;
* доступность;
* безопасность;
* сопровождаемость;
* совместимость.

Указание нефункциональных требования должно быть выражено посредством измеряемых критериев.

Например, 95% транзакций должно быть обработано менее чем за 1 секунду каждая, время простоя системы не должно превышать 1 минуту в сутки, и т.д.)

system downtime may not exceed 1 minute per day, > 30 day MTBF value, etc).

Для рассматриваемой системы сформировать не менее 10 нефункциональных требований. Требований представить в форме таблицы.

|  |  |
| --- | --- |
| **Нефункциональное требование** | **Тип требований** |
|  |  |

### ЛР6. Контрольные вопросы

1. Дать определение пользовательскому сценарию
2. Что такое базовый сценарий?
3. Что такое альтернативный сценарий?
4. Привести пример исключения (в контексте пользовательских сценариев)
5. Какая типовая структура описания функционального требования?
6. Привести пример функционального требования.
7. Привести пример нефункционального требования (в контексте производительности)
8. Привести пример нефункционального требования (в контексте надежности)

## 

## Лабораторная работа № 7. Концептуальная модель информационной системы

Время выполнения: 4 часа.

### ЛР7. Цель работы

приобрести навык формирования концептуальной модели информационной системы, включающей диаграмму потоков данных и компонентную диаграмму.

### ЛР7. Задания на лабораторную работу

1. На основе функциональной модели построить диаграмму потоков данных.
2. На основе функциональной модели, диаграммы потоков данных сформировать компонентную диаграмму.
3. Привести результаты (что сделано) и выводы (чему научились).

### ЛР7. Пояснения к выполнению лабораторной работы

Диаграммы потоков данных (Data Flow Diagrams - DFD) представляют собой иерархию функциональных процессов, связанных потоками данных. Задачей описания является формализация преобразования входных данных в выходные в рамках процесса, а также выявить отношения между процессами.

Для построения DFD традиционно используются две различные нотации, соответствующие методам Йордона-ДеМарко и Гейна-Сэрсона. Эти нотации незначительно отличаются друг от друга графическим изображением символов.

В соответствии с данными методами модель системы определяется как иерархия диаграмм потоков данных, описывающих асинхронный процесс преобразования информации от ее ввода в систему до выдачи потребителю. Практически любой класс систем успешно моделируется при помощи DFD-ориентированных методов. Они с самого начала создавались как средство проектирования информационных систем (тогда как SADT - как средство моделирования систем вообще) и имеют более богатый набор элементов, адекватно отражающих специфику таких систем (например, хранилища данных являются прообразами файлов или баз данных, внешние сущности отражают взаимодействие моделируемой системы с внешним миром).

По сути DFD представляет собой ориентированный граф, вершинами которого являются элементы системы или функции, а дуги – информационные потоки.

На основе дочерней диаграммы из лабораторной работы №5 построить диаграмму потоков данных.

Для описания физического представления системы используется компонентная диаграмма (или диаграмма компонентов). Диаграмма компонентов является одной из моделей архитектуры системы установив зависимости между программными компонентами, в роли которых может выступать исходный, бинарный и исполняемый код.

Для представления физических сущностей в языке UML применяется специальный термин – компонент (component).

Основными графическими элементами диаграммы компонентов являются компоненты, интерфейсы и зависимости между ними[[16]](#footnote-15).

Компонент обозначается прямоугольником со стереотипом компонента (текст <<компонент>>). Стереотип компонента обычно используется над именем компонента, чтобы не перепутать форму со значком класса.

Интерфейсы на диаграммах компонентов определяют порядок взаимодействия компонент между собой. Говорим, что компонент реализует интерфейс (окружность и сплошная линия). Если компонент использует интерфейс, то обозначается полукругом и сплошной линией. Фактически это можно расценивать, что один компонент предоставляет функцию, которая требуется другому компоненту.

Вы можете использовать компонентную диаграмму, когда хотите представить свою систему как компоненты и показать их взаимосвязь через интерфейсы. Это помогает получить представление о внедрении системы. Ниже приведены действия, которые можно выполнить при построении компонентной диаграммы[[17]](#footnote-16).

Шаг 1: выясните назначение диаграммы и идентифицируйте артефакты[[18]](#footnote-17), которые необходимо представить на диаграмме.

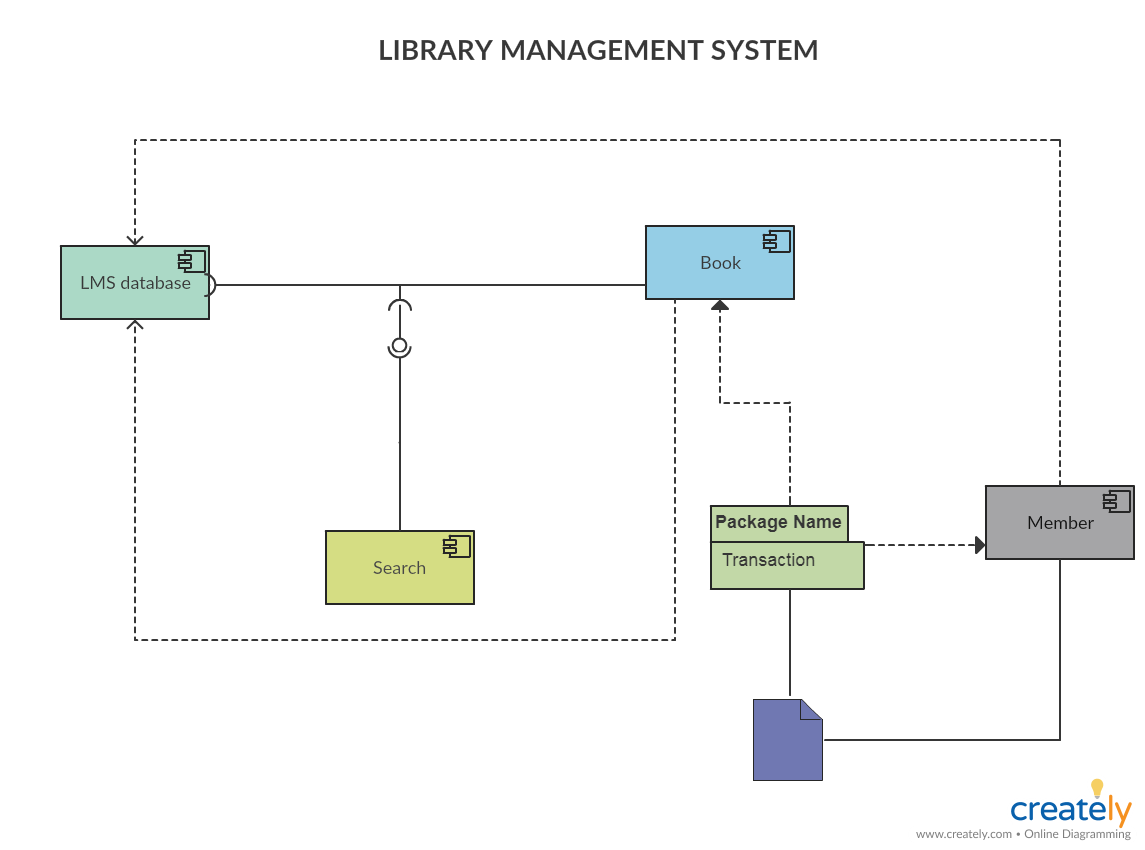
Шаг 2: По мере выяснения взаимосвязей между элементами, которые вы определили ранее, создайте ментальный макет своей компонентной диаграммы

Шаг 3: По мере того, как вы рисуете диаграмму, сначала добавьте компоненты, группируя их внутри других компонентов, как вам кажется подходящим

Шаг 4: Следующим шагом является добавление других элементов, таких как интерфейсы, классы, объекты, зависимости и т.д. в вашу компонентную диаграмму и ее завершение.

Шаг 5: Вы можете приложить примечания к различным частям вашей компонентной диаграммы, чтобы прояснить некоторые детали другим.

Пример диаграммы компонентов



### ЛР7. Контрольные вопросы

1. Дайте определение диаграммы потоков данных.
2. Приведите пример диаграммы потоков данных.
3. Что такое компонентная диаграмма?
4. Привести пример компонентной диаграммы.
5. Что такое интерфейс?
6. Дайте определение компонента.
7. Приведите пример интерфейса.

## 

## Лабораторная работа № 8. Испытания системы

Время выполнения: 4 часа.

### ЛР8. Цель работы

приобрести навык формирования плана испытаний системы.

### ЛР8. Задания на лабораторную работу

1. Разработать план проведения тестирования
2. Разработать метод функционального тестирования, состоящего из не менее 10 тестовых сценариев.
3. Разработать метод нефункционального тестирования.
4. Привести результаты (что сделано) и выводы (чему научились).

### ЛР8. Пояснения к выполнению лабораторной работы

Качество системы[[19]](#footnote-18) - это степень удовлетворения системой заявленных и подразумеваемых потребностей различных заинтересованных сторон, которая позволяет, таким образом, оценить достоинства.

Качество данных (data quality)[[20]](#footnote-19): Степень, с которой характеристики данных удовлетворяют заявленным и подразумеваемым требованиям при использовании в заданных условиях (ИСО/МЭК 25012).

Показатель внешнего качества программного обеспечения (external measure of software quality): Показатель степени, с которой программная продукция позволяет поведению (функционированию) системы удовлетворять заявленным и реализованным требованиям к этой системе, включая программное обеспечение при использовании в заданных условиях.

Обеспечение качества (QA — Quality Assurance) и контроль качества (QC — Quality Control) — это практики определяющие подходы для обеспечения и контроля качества системы.

QC (Quality Control) — совокупность действий и методов, в результате которого получение информации о том, что система соответствует требованиям к качеству.

QA (Quality Assurance) — совокупность действий и методов, целью которых является демонстрация соответствия системы требованиям к качеству.

К задачам обеспечения качества относятся[[21]](#footnote-20):

* проверка технических характеристик и требований к ПО;
* оценка рисков;
* планирование задач для улучшения качества продукции;
* подготовка документации, тестового окружения и данных;
* тестирование;
* анализ результатов тестирования, а также составление отчетов и других документов.

Валидация, аттестация (validation): Подтверждение (на основе представления объективных подтверждений) того, что требования, предназначенные для конкретного использования или применения, выполнены.

Примечание - Валидация в контексте жизненного цикла представляет собой совокупность действий, гарантирующих и обеспечивающих уверенность в том, что система способна реализовать свое предназначение, текущие и перспективные цели.

Другими словами валидация (validation) — это определение соответствия разрабатываемого ПО ожиданиям и потребностям пользователя, его требованиям к системе.

Верификация (verification): Подтверждение (на основе предоставления объективных подтверждений) того, что заданные требования полностью выполнены.

Примечание - Верификация в контексте жизненного цикла представляет собой совокупность действий по сравнению полученного результата жизненного цикла с требуемыми характеристиками для этого результата.

Другими словами верификация (verification) — это процесс оценки системы, чтобы понять, удовлетворяют ли результаты текущего этапа разработки условиям, которые были сформулированы в его начале.

Выделим два вида тестирования

* Функциональное тестирование (functional testing) — направлено на проверку корректности функционирования системы.
* Нефункциональное тестирование (non-functional testing) — тестирование атрибутов компонента или системы, не относящихся к функциональности.

Для рассматриваемой системы сформировать план проведения тестирования. План представить в виде таблицы.

|  |  |
| --- | --- |
| Пункт плана | Содержание |
| Введение (Introduction) |  |
| Объект тестирования (Test items) |  |
| Функции, которые будут протестированы (Features to be tested) |  |
| Функции, которые не будут протестированы (Features not to be tested) |  |
| Тестовые подходы (Approach) |  |
| Критерии прохождения тестирования (Item pass/fail criteria) |  |
| Критерии приостановления и возобновления тестирования (Suspension criteria and resumption requirements) |  |
| Результаты тестирования (Test deliverables) |  |
| Задачи тестирования (Testing tasks)[[22]](#footnote-21) |  |
| Ресурсы, необходимые для тестирования(Environmental needs); |  |
| Ответственность (Responsibilities)[[23]](#footnote-22); |  |
| Персонал и необходимая квалификация (Staffing and training needs); |  |
| Расписание (Schedule); |  |
| Риски и негативные последствия (Risks and contingencies); |  |
| Согласования (Approvals). |  |

Для тестирования функций вашей системы следует разработать метод тестирования (метод проверки работоспособности функций). Метод включает тестовые сценарии. Тестовый сценарий имеет следующую структуру:

* Предусловия (PreConditions) — список требований или условий, которые необходимо выполнить для приведения системы к состоянию пригодному для проведения тестирования.
* Шаги (Steps) — список действий, необходимых для проверки функционирования системы, позволяющие сделать вывод о том, что реализация функции системой удовлетворяет требованиям.
* Ожидаемый результат (Expected result) — ожидаемый результат после выполнения последовательности действий.

.

Пример.

Предусловия (PreConditions) — загружена страница аутентификации пользователя. Соединение с базой данных установлено.

Шаги (Steps) — (1) вводим в поле “логин” корректный логин, (2) вводим в поле “пароль” корректный пароль, соответствующий логину, (3) нажимаем на кнопку “Войти”.

Ожидаемый результат -- загружается страница личного кабинета пользователя с указанным логином.

Результаты представить в виде JSON файла

{

"test\_methos": "Метод тестирования",

"test\_case\_01": {

"PreConditions": {

"precondition\_description": "Описание предусловия",

"$ref": "test\_methos/test\_case\_01/PreConditions"

},

"Steps": {

"step\_01": {

"step\_name": "Шаг тестирования 1",

"$ref": "test\_methos/test\_case\_01/step\_01"

},

"step\_02": {

"step\_name": "Шаг тестирования 1",

"$ref": "test\_methos/test\_case\_01/step\_02"

}

},

"Expected\_result": {

"Expected\_results": "Описание ожидаемого результата",

"$ref": "test\_methos/test\_case\_01/Expected\_results"

}

}

}

Для нефункционального тестирование разработать методы тестирования оценки:

* производительности (определение стабильности и потребления ресурсов в условиях различных сценариев использования и нагрузок);
* масштабируемости (измеряет производительность системы при кратном увеличении пользователей)
* тестирование удобства использования;
* локализации (адаптации системы для определенной аудитории в соответствии с ее культурными особенностями)
* стрессовое тестирование (проверка, как система “справляется” с нарастающей нагрузкой).
* надёжности (проверка работоспособности системы при длительном тестировании с ожидаемым уровнем нагрузки)

Методы нефункционального тестирования представить в таблице.

|  |  |
| --- | --- |
| Метод тестирования... | Описание метода |
| производительности |  |
| масштабируемости |  |
| тестирование удобства использования |  |
| локализации |  |
| стрессовое тестирование |  |
| надёжности |  |

### ЛР8. Контрольные вопросы

1. Что такое качество системы?
2. Что такое качество данных?
3. Дайте определение понятию "обеспечение качества".
4. Дайте определение понятию "контроль качества".
5. Что такое валидация системы?
6. Что такое верификация системы?
7. Чем валидация отличается от верификации?
8. Дайте определение плану проведения тестирования?
9. Опишите структуру тестового сценария.
10. Перечислите что оценивается методом нефункционального тестирования.

## Оформление результатов выполнения лабораторной работы

### Основные требования

Результаты работы необходимо оформить в виде протокола, в котором написать:

* ФИО, группа;
* название работы;
* краткое описание задания, рассматриваемой системы и её подсистем;
* приложить результаты выполнения заданий (разработанные JSON-схемы, диаграммы и таблицы с подписями к ним и, при необходимости, сопроводительными комментариями).
* сформировать результаты и выводы

### Шаблон оформления протокола лабораторной работы

|  |  |
| --- | --- |
| Лабораторная работа |  |
| Группа |  |
| ФИО |  |
| Оценка: |  |

Цель лабораторной:

Название системы:

Решение задачи № 1:

….

Результаты и выводы.

Результаты

1. Выполнено …

Выводы:

## Перечень заданий на выполнение лабораторных работ

### Темы для выполнения задания

1. Программа управления задачами (Календарь, TODO.) для смартфона.
2. Портативное устройство считывания показаний со счётчика воды/электроэнергии с помощью камеры и устройства (Arduino, Raspberry Pi), собирающего данные.
3. Приложение дополненной реальности для визуализации в виде 3D-модели собранного конструктора[[24]](#footnote-23) по QR-коду на коробке.
4. Приложение, которое отправляет пользователю одну страницу в день о чем-то, чему он хочет научиться. Это может быть случайная страница из Википедии или из документации о какой-то технологии, видео с YouTube и т.д.
5. Надстройка для браузера для сокрытия спойлеров по заданным сериалам, книгам, фильмам на сайтах и в соц. сетях
6. Портативное устройство оценки качества воздуха в помещении на основании датчиков температуры, влажности, уровня CO2 и считывания этих же параметров на улице из погодных веб-сайтов (Arduino, Raspberry Pi).
7. Веб-приложение для обучения иностранному языку, которое заставляет пользователей переводить тексты с родного языка на изучаемый и отправлять их на проверку другим пользователям и наоборот.
8. Персональное хранилище мемов в виде надстройки в браузер, которая позволяет пользователю вставить в текст в мессенджере или соцсетях самостоятельно заданный (общепринятый или местечковый) мем (изображение, видео или аудиофайл) по коду или описанию.
9. Электронная телефонная записная книжка: аналог бумажной адресной книжки. Назначение: хранение информации о людях, поиск информации.
10. Электронный календарь. Назначение: фиксация событий, напоминания о событиях.
11. Калькулятор (стандартный с памятью на 1 ячейку)
12. Программа ежедневник (запись заметок, хранение локально в файле, возможность поиска по датам записи)
13. Текстовой редактор с функцией шифрования файлов (пароль задается при открытии файла)
14. Программа-диктофон.
15. Фото-галерея с простыми функциями редактирования (обрезка, повороты, яркость/контрастность)
16. Программа-будильник, с функцией запуска приложений.
17. Мобильное приложение для записи треков по GPS
18. Приложение учета семейного бюджета (доходы/расходы).
19. Приложение для персональных тренировок (занятия спортом).
20. Приложение “Афиша событий” <https://kudago.com/spb/>
21. Приложение учета калорий в потребляемой пище.
22. Приложение заказа билетов в кинотеатр.
23. Приложение отображения расписания занятий.
24. Приложение для курьерской доставки fast food.
25. Приложение для организации совместных закупок.
26. Приложение “Активный Гражданин” для сообщений о безобразиях в городе
27. Приложение фиксации ДТП (автоматизация европротокола)
28. Приложение для регистрации скидочных карт (менеджер скидочных карт)
29. Приложение отображения расписание мероприятий на конференции.
30. Приложение для управления освещением внутри дома (квартиры)
31. Приложение распознавания голоса с использованием Yandex Speech Kit (с микрофона поступает записанный голос, идет распознавание голоса с использованием Yandex Speech Kit, отображение распознанного текста.)
32. Приложение синтеза голоса с использованием Yandex Speech Kit (Ввод текста, Синтез голоса с использованием Yandex Speech Kit)
33. Запись события озвученного голосом в календарь Google
34. Запись события озвученного голосом в календарь Outlook
35. Запись события озвученного голосом в календарь Яндекс
36. Приложение для создания заметок на основе распознавания голоса
37. Генерация и отправка озвученного голосом электронного письма через Google
38. Генерация и отправка озвученного голосом электронного письма через Outlook
39. Генерация и отправка озвученного голосом электронного письма через Yandex
40. Программа управления компьютером (выключение , спящий режим)
41. Программа голосового запуска программ в Windows
42. Программа распознавания изображений животных
43. Приложение для распознавания эмоций человека
44. Приложение для распознавания животных по изображению
45. Веб-приложение для организации проведения в онлайне с друзьями какой-либо настольной игры (DnD и др.) c генерацией персонажей, интерактивными картами локаций и т.д.
46. Приложение определение ингредиентов в еде по картинке.
47. Мобильное приложение-помощник в приготовлении еды. Озвучивает голосом рецепт.
48. Умный ошейник для животных. Предназначен для отслеживания местоположения животного через мобильное приложение.
49. Программа определения пола человека по голосовому сигналу.
50. Программа распознавания дорожных знаков.

Выбор вариант задания N осуществляется по формуле:

N = M | M <= 50; N = M-50 | M > 50

M две последние цифры зачетной книжки.

# 

# Курсовой проект: методические указания по выполнению

## Общая информация

Курсовой проект по системной инженерии представляет собой законченную работу, включающую анализ существующих информационных систем и проектные решения по разработке компонента информационной системы.

Выбор темы курсового проекта осуществляется в соответствии с перечнем заданий и утверждается приказом ректора.

## Задачи курсового проекта

Курсовой проект состоит из 5 разделов/заданий по разделам.

1. выбор конкретной информационной системы в соответствии с темами и и краткое описание системы;
2. формирование перечня заинтересованных сторон[[25]](#footnote-24) информационной системы с построением матрицы влияния;
3. построение системной системной схемы;
4. описание сценариев использования системы;
5. построение структурной схемы;
6. представление информационной системы в виде графа состояний;
7. построение 2-х уровневой IDEF0 функциональной диаграммы системы;
8. выделение недостатков в существующей системе
9. построение системной системной схемы, включающей компоненты исключающие недостатки;
10. выводы.

В результате формируется пояснительная записка, содержащая результаты выполнения по разделам. Курсовой проект оформляется в соответствии с шаблоном и оценивается на защите. Оформляется задание на курсовой проект и пояснительная записка по шаблонам, приведённым на сайте ВолгГТУ: <http://umu.vstu.ru/umu-docs/uo/forms>

## Оценка курсового проекта

Курсовая работа оценивается максимум на 20 баллов. Каждый раздел -- 2 балла.

## Требования к разделам

1. выбор конкретной информационной системы в соответствии с темами и и краткое описание системы;
   1. в теме представлен класс систем, вам необходимо выбрать имеющуюся систему для которой есть описание (или которая имеет реализацию);
   2. в кратком описании привести:
      1. цели (количественные и качественные) создания системы;
      2. назначение системы;
2. формирование перечня заинтересованных сторон[[26]](#footnote-25) информационной системы с построением матрицы влияния;
   1. выделить не менее 4-х заинтересованных стороны (включая разработчиков, разработчиков смежных систем, эксплуатационный персонал, персонал сопровождения и разработчиков смежных систем)
3. построение системной схемы;
   1. в качестве целевой системы рассмотреть выбранную информационную систему;
   2. указать перечень обеспечивающий, использующих систем, систем в операционном окружении;
4. описание сценариев использования (пользовательских сценария) системы
   1. выделение ролей[[27]](#footnote-26);
   2. формирование не менее 5 сценариев использования для каждой роли;
5. построение структурной схемы;
   1. не менее 7 компонент[[28]](#footnote-27);
6. представление информационной системы в виде графа состояний;
   1. выделить 3 компонента информационной системы (например подсистемы)
   2. для каждого компонента реализовать граф состояний с матрицей переходов между состояниями
7. построение 2-х уровневой IDEF0 функциональной диаграммы системы;
   1. выделить главную функцию системы и построить IDEF0 диаграмму верхнего уровня для него;
   2. выполнить декомпозицию функций, указать не менее 4 функций на IDEF0 диаграмме ;
8. выделение недостатков в существующей информационной системе;
   1. не менее 5 недостатков;
   2. выполнить ранжирование недостатков по мере убывания влияния на качество работы системы
9. построение структурной системной схемы, включающей компоненты исключающие недостатки;
   1. построение выполнять на основе схемы 5;
   2. цветом выделить добавленные (измененные) компоненты.
10. Выводы.
    1. перечень результатов что сделано (например: “выбор информационная система... ”, “выделены недостатки … ”
    2. выводы по каждому пункту (ответ на вопрос и что дало решение вашей задачи). Например, “исходя из анализа недостатком можно прийти к выводу, что …. ”

## Темы

### Перечень тем (названия информационных систем)

1. Программно-измерительный комплекс оценки состояния животного (пример аналога [IoT Analysis of cattle activity "U-motion(R)" -- Short version](https://www.youtube.com/watch?v=xqhdlRmnL98))
2. Система предиктивного анализа отказов оборудования (пример аналога <https://www.youtube.com/watch?v=Sg3WhdY0Jb0>). Выбрать произвольное оборудование (техническую систему)
3. Система оценки эмоционального состояния человека-слушателя MOOC (см. пример аналога) <https://www.youtube.com/watch?v=7OcLybBZH40>
4. Система дистанционного измерения температуры тела человека, входящего в метро (пример прототипа <https://www.youtube.com/watch?v=1dghHV0dja4>)
5. Текстовый редактор для писателей, помогающий находить идеи при ступоре на основе машинного обучения на существующих литературных произведениях В дополнение к <https://ommwriter.com/> + <http://www.spacejock.com/yWriter6.html>
6. Система удаленного мониторинга состояния ветряных станций с помощью дрона (см. прототип системы <https://www.youtube.com/watch?v=L8yWygGUAx0&t=37s>)
7. Система автополива и удобрения растений, а также поддержания оптимальных условий микроклимата и освещения (интеллектуальная теплица) <https://create.arduino.cc/projecthub/londonium2021/automatic-indoor-vegetable-garden-9ea453>, <https://www.amazon.com/AS1001WH-Hydroponic-Growing-sensors-seasons/dp/B07QDSJ7NZ>
8. Система генерации автоматических отзывов на гостиницы (пример <https://www.youtube.com/watch?v=wTN-qKPu4c0>)
9. Надстройка в браузер, выдающая предложения к тексту, который пользователь пишет в соцсетях или мессенджерах, по исправлению его грамматических ошибок русского языка и улучшению его качества, чтобы текст больше был похож на строки классического произведения. На основе машинного обучения на классических литературных произведениях <https://www.kickstarter.com/projects/takahashi/talkmaster-improve-your-texting-skills>
10. Система семантической сегментации медицинских МРТ изображений и 3D реконструкции органов (пример аналога <https://www.youtube.com/watch?v=PcNqAVNCZrE>).
11. Система автоматического сбора яблок (<https://www.youtube.com/watch?v=mS0coCmXiYU>)

### 

### Порядок выбора темы

Выбор задания осуществляется в соответствии с таблицей 2, где M -- две последние цифры зачетной книжки

|  |  |
| --- | --- |
| **Номер темы** | **M** |
| 1 | 00,11,22,33,44,55,66,77,88,99 |
| 2 | 01,12,23,34,45,56,67,78,89, |
| 3 | 02,13,24,35,46,57,68,79,90, |
| 4 | 03,14,25,36,47,58,69,80,91, |
| 5 | 04,15,26,37,48,59,70,81,92 |
| 6 | 05,16,27,38,49,60,71,82,93, |
| 7 | 06,17,28,39,50,61,72,83,94, |
| 8 | 07,18,29,40,51,62,73,84,95, |
| 9 | 08,19,30,41,52,63,74,85,96, |
| 10 | 09,20,31,42,53,64,75,86,97, |
| 11 | 10,21,32,43,54,65,76,87,98, |

# 

# Литература

1. ГОСТ Р 57193-2016 Системная и программная инженерия. Процессы жизненного цикла систем. <https://docs.cntd.ru/document/1200141163>
2. ГОСТ Р ИСО/МЭК 25040-2014 Информационные технологии (ИТ). Системная и программная инженерия. Требования и оценка качества систем и программного обеспечения (SQuaRE). Процесс оценки <http://docs.cntd.ru/document/1200111327>
3. ГОСТ Р ИСО/МЭК 25010-2015 Информационные технологии. Cистемная и программная инженерия. Требования и оценка качества систем и программного обеспечения (SQuaRE). Модели качества систем и программных продуктов <https://docs.cntd.ru/document/1200121069>
4. ГОСТ Р ИСО/МЭК 25021-2014 Информационные технологии. Системная и программная инженерия. Требования и оценка качества систем и программного обеспечения (SQuaRE). Элементы показателя качества <https://docs.cntd.ru/document/1200111326>
5. ГОСТ Р 56920-2016/ISO/IEC/IEEE 29119-1:2013 СИСТЕМНАЯ И ПРОГРАММНАЯ ИНЖЕНЕРИЯ Тестирование программного обеспечения
6. ГОСТ Р 51167-98 Качество служебной информации. Графические модели технологических процессов переработки данных <https://docs.cntd.ru/document/1200025651>
7. ГОСТ Р ИСО/МЭК 11179-3-2012 Информационная технология, Регистры метаданных (РМД), Часть 3, Метамодель регистра и основные атрибуты <https://docs.cntd.ru/document/1200098406>
8. Управление требованиями к IT-проектам <https://habr.com/ru/post/114571/>
9. ГОСТ Р 59194-2020 Управление требованиями, Основные положения <http://docs.cntd.ru/document/573219705>
10. ГОСТ Р 57100-2016/ISO/IEC/IEEE 42010:2011 Системная и программная инженерия. Описание архитектуры. <https://docs.cntd.ru/document/1200139542>
11. Astah https://astah.net/,
12. Umbrello https://umbrello.kde.org/,
13. Draw.io https://www.draw.io/.
14. UML 2 State Machine Diagramming Guidelines <http://agilemodeling.com/style/stateChartDiagram.htm>
15. What is State Machine Diagram? <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-state-machine-diagram/>
16. JSON Formatter <https://jsonformatter.curiousconcept.com/>
17. JSON Online Editor <https://jsoneditoronline.org/>
18. Основы пользовательских историй. Часть 1. Введение <https://habr.com/ru/company/luxoft/blog/82066/>
19. Основы UML — диаграммы использования (use-case) <https://pro-prof.com/archives/2594>
20. UML <http://khpi-iip.mipk.kharkiv.edu/library/case/leon/index.html>
21. Нотация и семантика языка UML <https://intuit.ru/studies/courses/32/32/info>
22. Описание нотации IDEF0 <https://micro-solution.ru/bp-know/regulation/IDEF0>
23. Методология IDEF0 <https://www.sites.google.com/site/anisimovkhv/learning/pris/lecture/tema6/tema6_2>
24. Перевод стандарта IDEF0 на русском языке <https://habr.com/ru/post/515466/>
25. ГЛАВА 10 Диаграмма компонентов (component diagram) <http://www.telenir.net/uchebniki/samouchitel_uml/p10.php>
26. Фундаментальная теория тестирования <https://habr.com/ru/post/549054/>
27. Matlab Systems Engineering <https://www.youtube.com/playlist?list=PLn8PRpmsu08owzDpgnQr7vo2O-FUQm_fL>
28. MIT 16.842 Fundamentals of Systems Engineering, Fall 2015 View the complete course: <http://ocw.mit.edu/16-842F15> Instructor: Olivier de Weck <https://www.youtube.com/playlist?list=PLUl4u3cNGP60jIMmB53zl6awCKMnABhYx>
29. System engineering lectures <https://www.youtube.com/playlist?list=PLelAq9xzEDXC5ikiGLE3CklnK5_VRW_bf> Dr. Orlando J. Hernandez <https://www.youtube.com/user/adashwit/videos>
30. SEBok Part 3.1 - 3.2 <https://www.sebokwiki.org/wiki/Introduction_to_Life_Cycle_Processes>
31. <https://www.mitre.org/publications/systems-engineering-guide/se-life-cycle-building-blocks>
32. <https://www.mitre.org/publications/systems-engineering-guide/enterprise-engineering/systems-of-systems/systems-engineering>
33. <https://www.tonex.com/training-courses/introduction-systems-engineering-lifecycle-processes/>
34. <http://www.agilemodeling.com/artifacts/stateMachineDiagram.htm>
35. <http://agilemodeling.com/style/stateChartDiagram.htm>
36. <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-state-machine-diagram/>
37. <https://www.projectmanager.com/software/use-cases/sdlc>
38. <https://existek.com/blog/sdlc-models/>
39. <https://melsatar.blog/2012/03/15/software-development-life-cycle-models-and-methodologies/>

1. Выбор системы осуществляется из перечня заданий. [↑](#footnote-ref-0)
2. система которая выбрана в первом задании. [↑](#footnote-ref-1)
3. Выбор системы осуществляется из перечня заданий. [↑](#footnote-ref-2)
4. т.е. когда система уже воплощена и функционирует в соответствии с замыслом. [↑](#footnote-ref-3)
5. Описывать лучше в специализированном редакторе, например, Visual Studio, Notepad++, Visual Studio Code, Sublime text. Выполнить валидацию JSON-файла можно в специальных сервисах JSON-форматтеров, например: <https://jsonformatter.curiousconcept.com/> или <https://jsoneditoronline.org/> [↑](#footnote-ref-4)
6. стартовая страница ya.ru [↑](#footnote-ref-5)
7. Если замечаний или рекомендаций нет, изменения в структурную схему не вносятся. [↑](#footnote-ref-6)
8. Качественная или количественная [↑](#footnote-ref-7)
9. Основы пользовательских историй. Часть 1. Введение: <https://habr.com/ru/company/luxoft/blog/82066/> [↑](#footnote-ref-8)
10. Подробно о диаграмме использования <https://pro-prof.com/archives/2594> , <http://khpi-iip.mipk.kharkiv.edu/library/case/leon/gl4/gl4.html> [↑](#footnote-ref-9)
11. будем рассматривать возможности как функции системы [↑](#footnote-ref-10)
12. <https://intuit.ru/studies/courses/32/32/lecture/1004> [↑](#footnote-ref-11)
13. Полезный материал: <https://micro-solution.ru/bp-know/regulation/IDEF0> и еще один <https://www.sites.google.com/site/anisimovkhv/learning/pris/lecture/tema6/tema6_2> [↑](#footnote-ref-12)
14. Перевод стандарта IDEF0 на русском языке <https://habr.com/ru/post/515466/> [↑](#footnote-ref-13)
15. Данная информация в основном предназначена, если ожидаемый результат программный продукт. В данном случае пользовательские сценарии могут быть расширены описанием API. [↑](#footnote-ref-14)
16. Подробное описание компонентной диаграммы: <http://www.telenir.net/uchebniki/samouchitel_uml/p10.php> [↑](#footnote-ref-15)
17. Взято из источника: <https://creately.com/blog/ru/uncategorized-ru/%D1%83%D1%87%D0%B5%D0%B1%D0%BD%D0%BE%D0%B5-%D0%BF%D0%BE%D1%81%D0%BE%D0%B1%D0%B8%D0%B5-%D0%BF%D0%BE-%D0%BA%D0%BE%D0%BC%D0%BF%D0%BE%D0%BD%D0%B5%D0%BD%D1%82%D0%BD%D0%BE%D0%B9-%D0%B4%D0%B8%D0%B0%D0%B3/> [↑](#footnote-ref-16)
18. артефакты для информационной системы: файлы, документы и т.д. [↑](#footnote-ref-17)
19. Определение взято из <https://docs.cntd.ru/document/1200121069> [↑](#footnote-ref-18)
20. Определение взято из <https://docs.cntd.ru/document/1200111326> [↑](#footnote-ref-19)
21. Фундаментальная теория тестирования <https://habr.com/ru/post/549054/> [↑](#footnote-ref-20)
22. Типы тестирования (в работе ограничимся функциональным и нефункциональным тестированием) [↑](#footnote-ref-21)
23. описывается область ответственности персонала (или автомата), которые реализуют тестирование [↑](#footnote-ref-22)
24. Например LEGO [↑](#footnote-ref-23)
25. не забываем разработчиков [↑](#footnote-ref-24)
26. не забываем разработчиков [↑](#footnote-ref-25)
27. ролей может быть меньше, чем заинтересованных лиц [↑](#footnote-ref-26)
28. меньше не получится учитывая сложность системы [↑](#footnote-ref-27)