**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ**

**РОССИЙСКОЙ ФЕДЕРАЦИИ**

федеральное государственное бюджетное образовательное учреждение высшего образования «Казанский национальный исследовательский технический университет им. А. Н. Туполева — КАИ»

(КНИТУ-КАИ)

Институт компьютерных технологий и защиты информации

Кафедра «Системы автоматизированного проектирования»

Направление 09.03.01- «Информатика и вычислительная техника»

Профиль «Системы автоматизированного проектирования

(электронные средства)»

**К защите допустить**

**Зав. каф.**

« » 2020 г.

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

на тему «Прогнозирование надежности сложных технических систем»

ОБУЧАЮЩИЙСЯ Хабибуллин А.Э.

*(личная подпись)*

РУКОВОДИТЕЛЬ д. т. н., профессор Чермошенцев С.Ф.

*(личная подпись)*

Казань, 2020 год

**MINISTRY OF SCIENCE AND HIGHER EDUCATION**

**OF THE RUSSIAN FEDERATION**

Federal State-Funded Educational Institution of Higher Education «Kazan National Research Technical University named after A. N. Tupolev - KAI»

(KNRTU - KAI)

Institute of Computer Technology and Information Security

Department of Computer Aided Design Systems

Training profile « Computer Aided Design Systems (electronic equipments)»

**Allow to protect**

**Head of the chair**

« » 2020 г.

GRADUAL QUALIFICATION WORK

on the topic «Reliability prediction of technical systems»

STUDENT A.E. Khabibullin

*(personal signature)*

ADVISER Doctor of Technical Sciences, Professor

Chermoshentsev S.F.

*(personal signature)*

Kazan, 2020 year

**Календарный план**

|  |  |  |  |
| --- | --- | --- | --- |
| *№*  *п/п* | Наименование этапов  выпускной квалификационной работы | Срок выполнения  этапов выпускной квалификационной работы | Примечание |
| *1.* | *Рассмотрение современных тенденций и проблем прогнозирования надежности сложных технических систем* | *01.11.19 – 15.11.19* |  |
| *2.* | *Разработка функциональных и поведенческих моделей (IDEF0 и IDEF3) проектной процедуры прогнозирования надежности сложных технических систем* | *16.11.19 – 29.11.19* |  |
| *3.* | *Обзор систем автоматизаци проектной процедуры прогнозирования надежности сложных технических систем* | *30.11.19 – 13.12.19* |  |
| *4.* | *Постановка цели и задач автоматизации проектной процедуры прогнозирования надежности сложных технических систем* | *14.12.19 – 28.12.19* |  |
| *5.* | *Содержательная и математическая постановка задачи прогнозирования надежности сложных технических систем* | *29.12.19 – 18.01.20* |  |
| *6.* | *Обзор методов и алгоритмов решения задачи прогнозирования надежности сложных технических систем* | *19.01.20 – 31.01.20* |  |
| *7.* | *Изучение и описание алгоритма решения задачи прогнозирования надежности сложных технических систем* | *01.02.20 – 14.02.20* |  |
| *8.* | *Разработка модели потока данных (DFD) и концептуальной модели базы данных* | *15.02.20 – 28.02.20* |  |
| *9.* | *Разработка логической модели базы данны* | *29.02.20 – 20.03.20* |  |
| *10.* | *Физическое проектирование базы данных* | *21.03.20 – 17.04.20* |  |
| *11.* | *Разработка архитектуры программы для автоматизации проектной процедуры прогнозирования надежности сложных технических систем* | *18.04.20 – 01.05.20* |  |
| *12.* | *Разработка пользовательского интерфейса программы для автоматизации прогнозирования надежности сложных технических систем.* | *02.05.20 – 15.05.20* |  |
| *13.* | *Решение задачи прогнозирования надежности сложных технических систем.на контрольных примерах* | *16.05.20 – 29.05.20* |  |
| *14.* | *Подготовка к защите выпускной квалификационной работы* | *30.05.20 – 08.06.20* |  |

Обучающийся \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**АННОТАЦИЯ**

Данная работа посвящена разработке подсистемы для прогнозирования надежности сложных технических систем. Проанализирована предметная область, разработано математическое и информационное обеспечение. Итогом работы является программа прогнозирования надежности сложных технических систем любой структуры с помощью логико-вероятностного метода.

**ANNOTATION**

This work is devoted to the development of a subsystem for predicting the reliability of complex technical systems. The subject area is analyzed, mathematical and information support is developed. The result of this work is a program for predicting the reliability of complex technical systems of any structure using the logical-probabilistic method.

**СОДЕРЖАНИЕ**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Введение……………………………………………………………………. | | 11 |
| 1. | Анализ проектной процедуры прогнозирования надежности сложных технических систем ……………………………………………………….. | | 12 |
|  | 1.1. | Современные тенденции и проблемы прогнозирования надежности сложных технических систем ………………………... | 12 |
|  | 1.2. | Разработка функциональных и поведенческих моделей (IDEF0 и IDEF3) проектной процедуры прогнозирования надежности сложных технических систем ………………….…………………... | 13 |
|  |  | 1.2.1. Функциональная модель проектной процедуры IDEF0……. | 14 |
|  |  | 1.2.2. Модель проектной процедуры IDEF3………………………... | 21 |
|  | 1.3. | Обзор систем автоматизации проектной процедуры прогнозирования надежности сложных технических систем......... | 22 |
|  | 1.4. | Цель и задачи автоматизации проектной процедуры прогнозирования надежности сложных технических систем ….… | 24 |
|  |  | Выводы по главе 1…………………………………………………... | 25 |
| 2. | Разработка математического обеспечения для автоматизации проектной процедуры прогнозирования надежности сложных технических систем………………………………………………………... | | 26 |
|  | 2.1. | Содержательная и математическая постановка задачи прогнозирования надежности сложных технических систем ...….. | 26 |
|  | 2.2. | Методы и алгоритмы решения задачи прогнозирования надежности сложных технических систем........................................ | 29 |
|  | 2.3. | Разработка алгоритма решения задачи прогнозирования надежности сложных технических систем …………...…………… | 33 |
|  |  | Выводы по главе 2…………………………………………...……… | 38 |
| 3. | Разработка информационного обеспечения для автоматизации проектной процедуры прогнозирования надежности сложных технических систем …..………………………………………………..…. | | 39 |
|  | 3.1. | Разработка модели потока данных (DFD) и концептуальной модели базы данных ………………………………………….…….. | 39 |
|  | 3.2. | Разработка логической модели базы данных……………………… | 42 |
|  | 3.3. | Физическое проектирование базы данных………………………… | 43 |
|  |  | Выводы по главе 3…………………………………………………… | 44 |
| 4. | Разработка программного обеспечения для автоматизации проектной процедуры прогнозирования надежности сложных технических систем | | 45 |
|  | 4.1. | Архитектура программы для автоматизации проектной процедуры прогнозирования надежности сложных технических систем……………………………………………………………….... | 45 |
|  |  | 4.1.1. Анализ функциональных требований к программе………… | 45 |
|  |  | 4.1.2. Описание архитектуры программы………………………….. | 56 |
|  | 4.2. | Разработка пользовательского интерфейса программы для автоматизации прогнозирования надежности сложных технических систем…………………………………………………. | 66 |
|  | 4.3. | Примеры решения задачи прогнозирования надежности сложных технических систем…………………………………………………. | 80 |
|  |  | Выводы к главе 4…………………………………………………….. | 85 |
|  | Заключение……………………………………………………………….... | | 86 |
|  | Список литературы………………………………………………………… | | 88 |
|  | Приложение 1. Глоссарий…………………………………………………. | | 91 |
|  | Приложение 2. Листинг программы……………………………………… | | 93 |

**ВВЕДЕНИЕ**

Задачи по прогнозированию надежности в настоящее время актуальны как никогда, поскольку они дают ответ на вопрос о целесообразности дальнейших затрат, необходимых на обработку технологии и производство сложных технических систем.

Адекватное моделирование надежности сложных технических систем может быть реализовано только с помощью автоматизации. Основными для анализа надежности являются структурные методы, позволяющие представить систему в виде структурной схемы, описывающей логические соотношения между состояниями элементов и системы в целом с учетом структурно-функциональных связей и взаимодействия элементов.

В рамках данной выпускной квалификационной работы будут исследованы методы расчета надежности сложных технических систем, а также будет реализовано программное обеспечение, позволяющее проводить анализ надежности с помощью логико-вероятностного метода.

**ГЛАВА 1. Анализ проектной процедуры**

**1.1. Современные тенденции и проблемы прогнозирования надежности сложных технических систем**

*Надежность* - это свойство изделия сохранять в течение заданного времени в пределах установленных норм значения функциональных параметров при определенных условиях [2]. Надежность - комплексное свойство изделия, для количественного описания которого используют несколько групп показателей:

* показатели безотказности;
* показатели ремонтопригодности;
* показатели долговечности;
* показатели сохраняемости;
* комплексные показатели надежности.

По мере технического развития происходит усложнение изделий приборостроения. Главное противоречие в развитии современной техники заключается в том, что если не предпринимать необходимые меры по повышению надёжности, то чем сложнее и точнее работа техники, тем менее она надёжна. Отсюда следует, что решение проблемы надежности - это не только важная техническая задача, но и большая экономическая. [10]

Выпускаемые промышленностью изделия, в том числе и электронные средства (ЭС) предназначены для выполнения заданных функций в течение определенного времени. Способность изделий безотказно и эффективно функционировать характеризуются показателями надежности, отражающими определенные свойства изделия. В отличие от любых физических величин надежность не может быть непосредственно измерена, а может быть только количественно оценена или предсказана, так как это свойство изделий относится к их предстоящей эксплуатации, условия которой разнообразны и мало предсказуемы. Кроме того, не существует единственного критерия, который полностью бы характеризовал надежность сложной системы, в связи с ее многофункциональностью. Это явилось одной из важнейших причин возникновения нового научного направления теории надежности. [10]

*Теория надежности -* наука, изучающая закономерности отказов технических систем. [16]

Основными *объектами*ее изучения являются:

* критерии надежности технических систем различного назначения;
* методы анализа надежности в процессе проектирования и эксплуатации технических систем;
* методы синтеза технических систем;
* пути обеспечения и повышения надежности техники;
* научные методы эксплуатации техники, обеспечивающие ее высокую надежность.

*Особенности*этой дисциплины таковы:

* теория надежности — общетехническая дисциплина;
* математическое моделирование — основа изучения дисциплины;
* комплексный характер;
* высокая значимость и глубокая связь с другими техническими предметами;
* трудность моделирования и изучения процессов, протекающих в сложных технических системах (в смысле их надежности).

В настоящее время теория надежности является одной из фундаментальных научных дисциплин, математический аппарат которой составляют теория вероятностей и математическая статистика. [20]

**1.2. Разработка функциональных и поведенческих моделей (IDEF0 и IDEF3) проектной процедуры прогнозирования надежности сложных технических систем.**

**1.2.1. Функциональная модель проектной процедуры IDEF0**

На рис. 1 – рис. 4 представлена IDEF0 диаграмма решения задачи прогнозирования надежности сложных технических систем и ее декомпозиция.

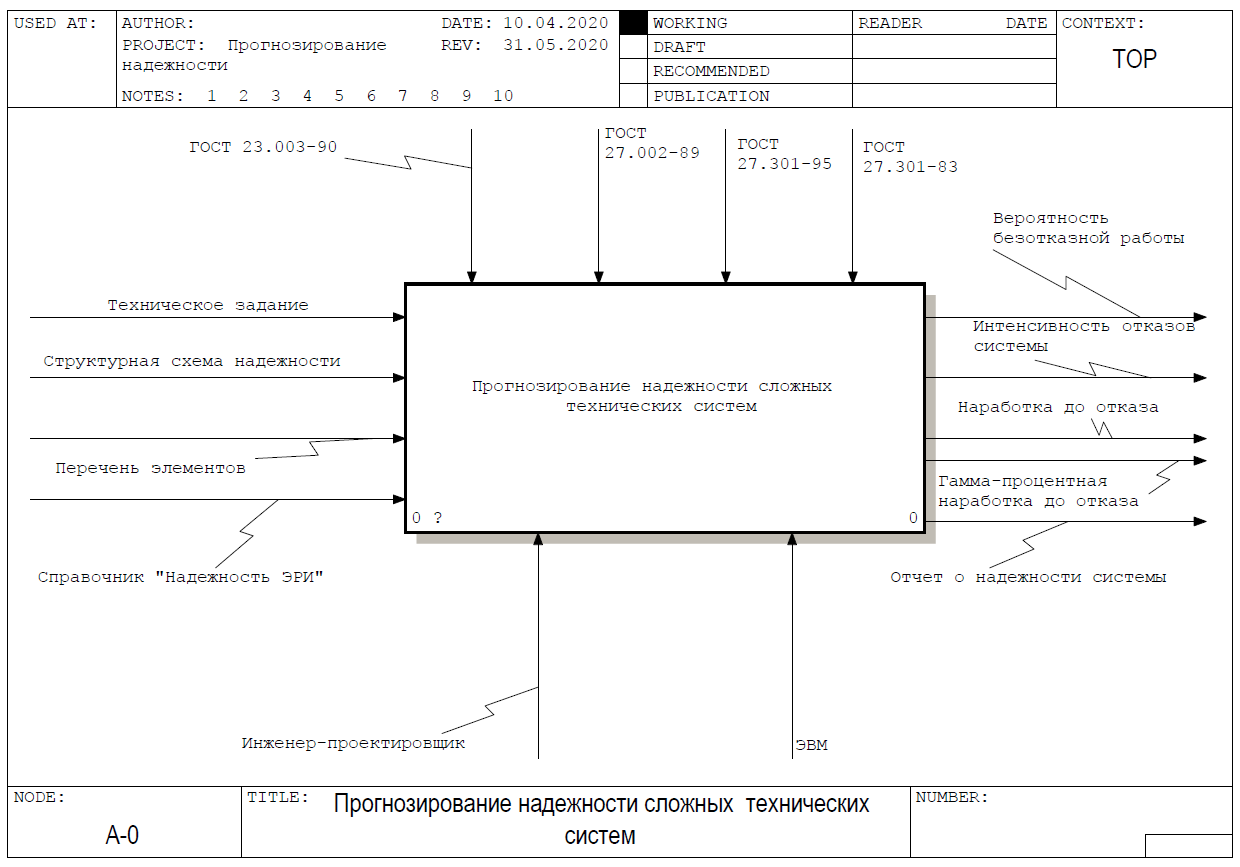


Рис. 1. Диаграмма IDEF0 A-0

Входными данными в IDEF A-0 диаграмме являются:

* техническое задание (ТЗ) - документ, содержащий основные требования по надежности;
* структурная схема надежности – графическое представление структуры системы с точки зрения надежности;
* перечень элементов с необходимыми для расчета надежности параметрами каждого элемента. В качестве параметров могут выступать интенсивность отказов (базовая или эксплуатационная) и вероятность безотказной работы;
* справочник "Надежность электрорадиоизделий", содержащий все необходимые сведения о надежности электрорадиоизделий (ЭРИ) отечественного производства. Используется в том случае, если в качестве элемента выступает ЭРИ и к нему в техническом задании или перечне элементов не указаны базовые показатели надежности.

Выходными данными в IDEF A-0 диаграмме являются:

* интенсивность отказов системы;
* вероятность безотказной работы;
* наработка до отказа;
* гамма процентная наработка до отказа;
* отчет о надежности системы, в который заносится вся имеющаяся информация о надежности технической системы.

Описание вышеперечисленных показателей надежности представлено в разделе 3.1.

Управление процессом осуществляется при помощи нормативных документов и персонала:

* ГОСТ 27.002-89 «Надежность в технике (ССНТ). Основные понятия. Термины и определения»;
* ГОСТ 27.301-95 «Надежность в технике. Расчет надежности. Основные положения», содержащий основные положения по расчету надежности в технике;
* ГОСТ 27.301-83 «Надежность в технике. Прогнозирование надежности изделий при проектировании. Общие требования»;
* ЭВМ, с помощью которой будет производиться расчет надежности;
* инженер-проектировщик - специалист, отвечающий за сбор исходных данных по проекту, производящий технико-экономические расчеты, увязку конструктивных составляющих и управленческих решений различных разделов, а также разработку чертежной документации.

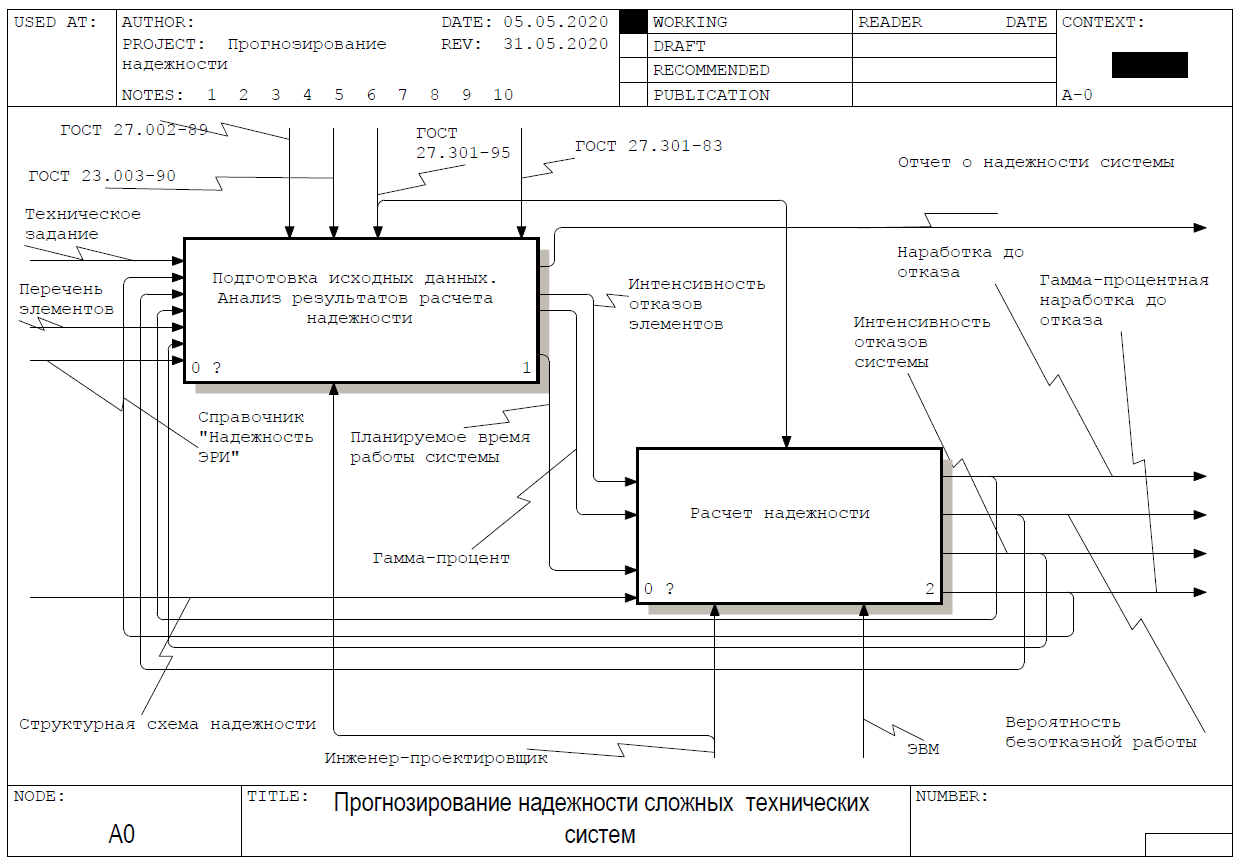


Рис. 2. Диаграмма IDEF0 A0

Происходит подготовка исходных данных. Интенсивность отказов элементов берется из перечня элементов или справочника «Надежность электрорадиоизделий», планируемое время работы системы и гамма-процент – из технического задания. Далее осуществляется автоматизированный расчет надежности с использованием логико-вероятностного метода. Результатом расчета являются количественные показатели надежности. В конечном счете, осуществляется анализ рассчитанных показателей надежности. Затем эти показатели используются для оформления отчета о надежности технической системы.

Входными данными в IDEF A0 диаграмме являются:

* техническое задание;
* структурная схема надежности;
* перечень элементов.
* справочник "Надежность электрорадиоизделий";

Выходными данными в IDEF A0 диаграмме являются:

* интенсивность отказов системы;
* вероятность безотказной работы;
* наработка до отказа;
* гамма процентная наработка до отказа;
* планируемое время работы системы – время, указанное в ТЗ, в течение которого устройство должно исправно работать;
* интенсивность отказов элементов. Берется перечня элементов. Если элементом является ЭРИ, то интенсивность отказов можно взять из справочника «Надежность электрорадиоизделий»;
* гамма-процент;
* отчет о надежности системы.

Управление процессом осуществляется при помощи нормативных документов и персонала:

* ЭВМ;
* инженер-проектировщик.

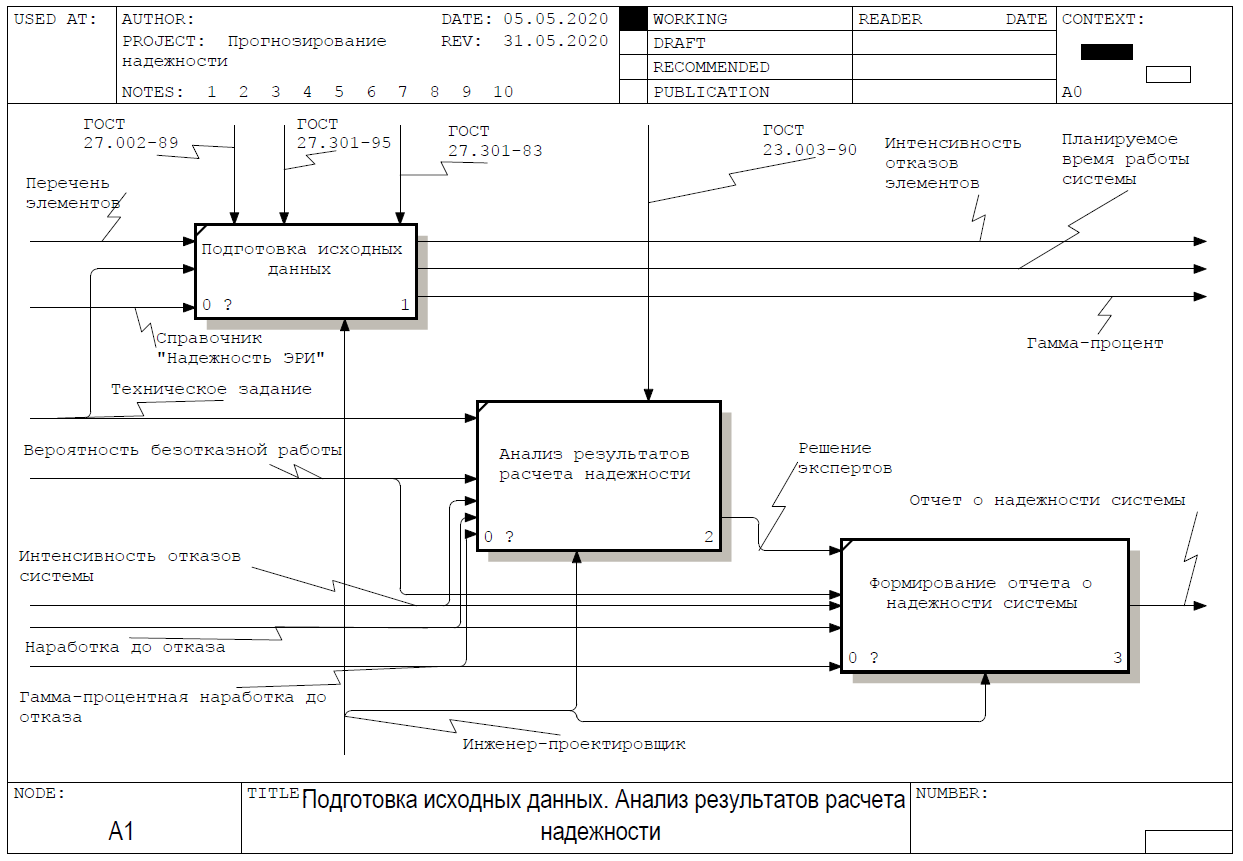


Рис. 3. Диаграмма IDEF0 A1

Сначала осуществляется подготовка исходных данных, после чего эти данные используются для автоматизированного расчета надежности технической системы. После расчета надежности осуществляется анализ полученных результатов. В случае, если рассчитанные показатели надежности не удовлетворяют требованиям ТЗ, в отчет о надежности, помимо самих показателей надежности, записываются рекомендации по повышению надежности технической системы.

Входными данными в IDEF A1 диаграмме являются:

* перечень элементов;
* техническое задание;
* справочник «Надежность электрорадиоизделий»;
* вероятность безотказной работы;
* интенсивность отказов;
* наработка до отказа;
* гамма-процентная наработка до отказа;

Выходными данными в IDEF A1 диаграмме являются:

* интенсивность отказов элементов;
* планируемое время работы системы;
* гамма-процент;
* решение экспертов, основанное на анализе полученных показателей надежности. В случае, если рассчитанные показатели надежности не удовлетворяют требованиям ТЗ, в отчет о надежности, помимо самих показателей надежности, записываются рекомендации по повышению надежности технической системы;
* отчет о надежности системы.

Управление процессом осуществляется при помощи нормативных документов и персонала:

* инженер-проектировщик.

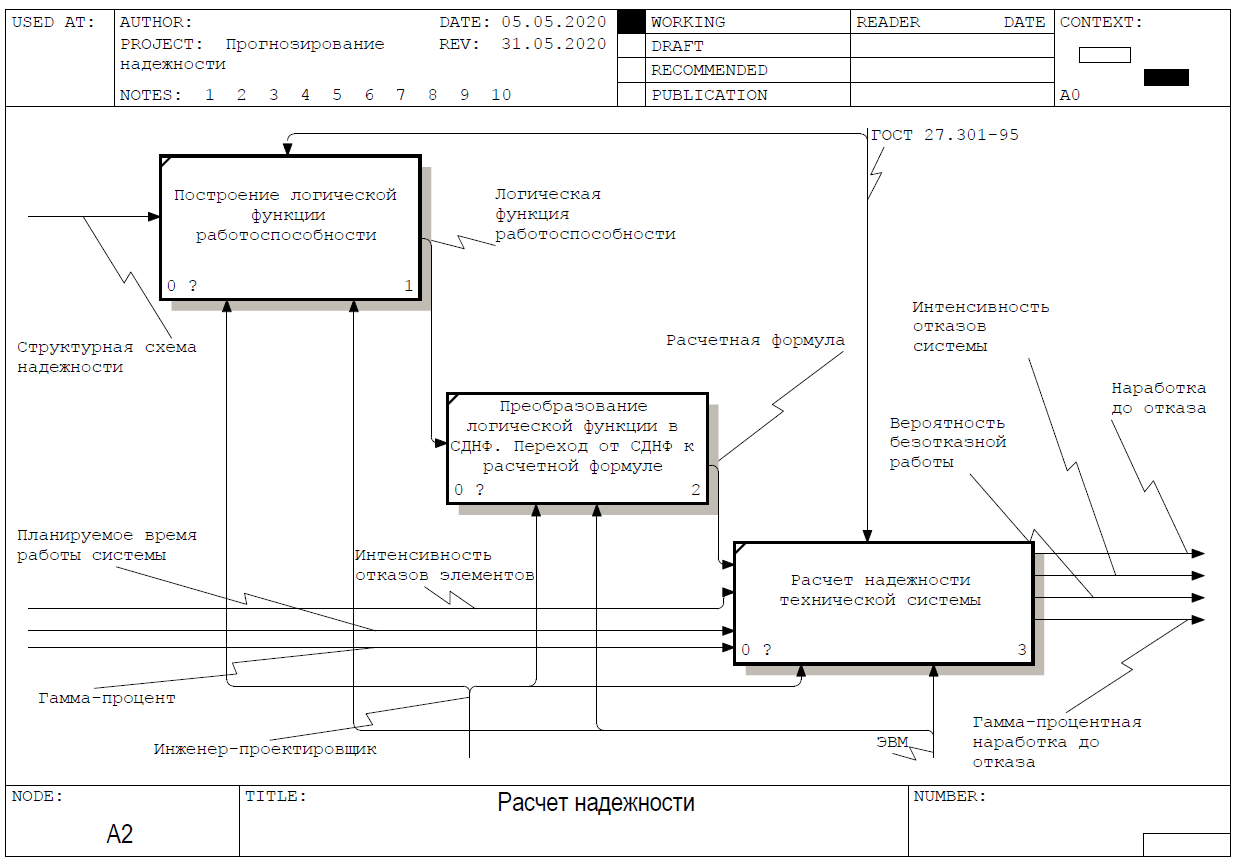


Рис. 4. Диаграмма IDEF0 A2

На основе анализа структурной схемы надежности технической системы, формируется логическая функция работоспособности системы. Инженер-проектировщик, используя ЭВМ, вводит эту функцию в программе. Логическая функция работоспособности преобразуется в совершенную дизъюнктивную нормальную форму (СДНФ). Происходит переход от СДНФ к расчетной формуле путем замены логических операций на алгебраические. Производится автоматизированный расчет надежности с помощью логико-вероятностного метода, основанный на введенной логической функции работоспособности, времени работы системы, интенсивности отказов элементов и гамма-проценте.

Входными данными в IDEF A2 диаграмме являются:

* интенсивность отказов элементов;
* планируемое время работы системы;
* гамма-процент;
* структурная схема надежности.

Выходными данными в IDEF A2 диаграмме являются:

* интенсивность отказов системы;
* вероятность безотказной работы;
* наработка до отказа;
* гамма процентная наработка до отказа;
* логическая функция работоспособности – описание структуры надежности сложных технических систем средствами математического аппарата бинарной алгебры логики;
* расчетная формула. Получают путем преобразования логической функции работоспособности в совершенную дизъюнктивную нормальную форму и замены в ней логических операций на алгебраические.

Управление процессом осуществляется при помощи нормативных документов и персонала:

* справочник [9];
* ЭВМ;
* инженер-проектировщик.

**1.2.2. Модель проектной процедуры IDEF3**

На рис. 5 представлена IDEF3 диаграмма решения задачи прогнозирования надежности сложных технических систем.

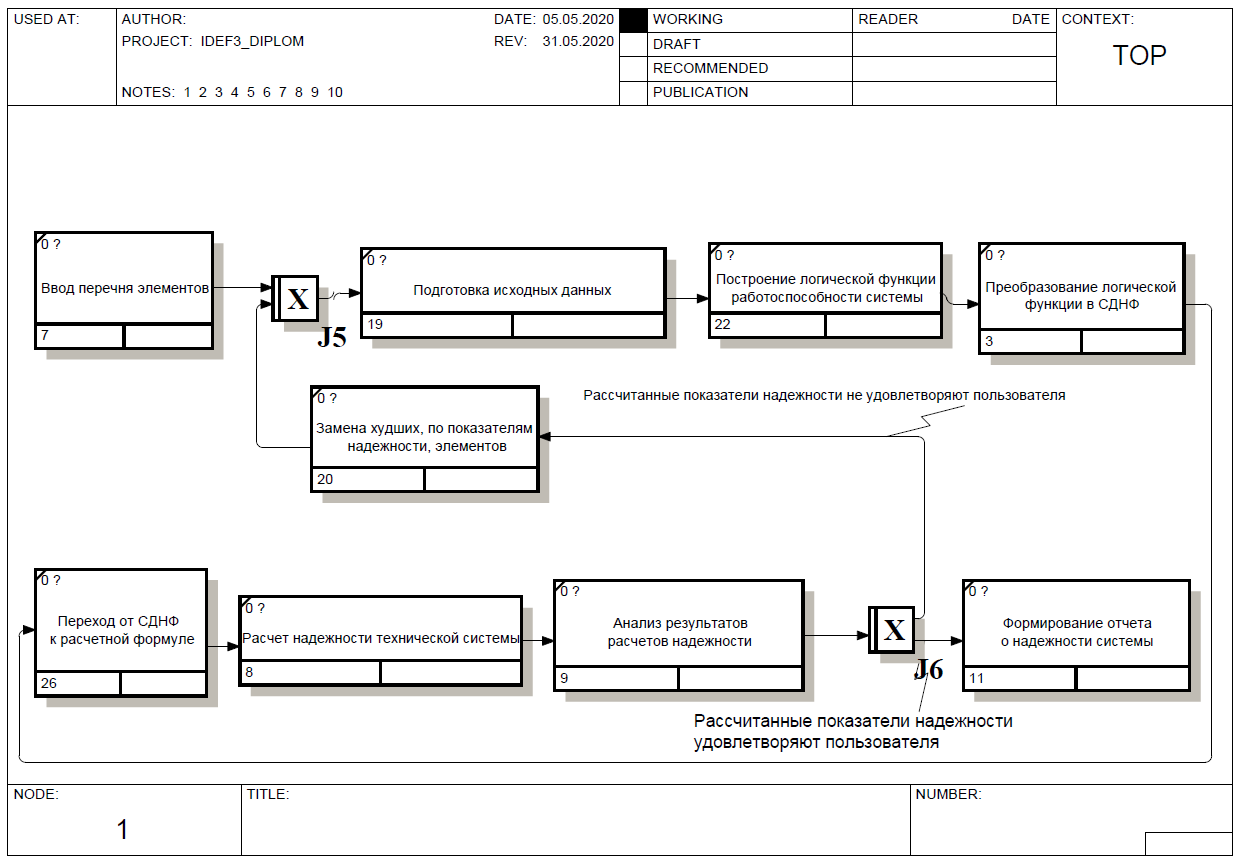


Рис. 5. IDEF3 диаграмма

Сначала пользователь осуществляет ввод перечня элементов, входящих в техническую систему. Далее он осуществляет подготовку исходных данных: интенсивность отказов элементов, планируемое время работы технической системы и гамма-процент. Затем пользователь вводит логическую функцию работоспособности системы. Логическая функция работоспособности преобразуется в совершенную дизъюнктивную нормальную форму. Происходит переход от СДНФ к расчетной формуле путем замены логических операций на алгебраические. Производится расчет надежности с помощью логико-вероятностного метода на основе введенной логической функции работоспособности. После расчета пользователю необходимо решить, удовлетворяют ли его рассчитанные показатели надежности. Если результаты расчета удовлетворительные, пользователь формирует отчет формата файлов Excel, в ином случае пользователь заменяет элементы с наихудшими показателями надежности на высоконадежные аналоги.

**1.3. Обзор систем автоматизации проектной процедуры прогнозирования надежности сложных технических систем**

Рассмотрим существующие программные комплексы по расчету надежности сложных технических систем.

Наиболее известными среди зарубежных программных комплексов являются: [24]

* RELEX (Relex software Corporation;
* A.L.D. Group;
* Risk Spectrum;
* ISOGRAPH.

Программные комплексы RELEX и Risk Spectrum позволяют проводить расчет надежности с помощью логико-вероятностного метода. Risk Spectrum применяется в основном для объектов атомной энергетики: он используется более чем на 50% атомных станций мира. [24]

Фирма A.L.D. Group объединяет две компании, специализирующие в области логистики и оценки надежности: SoHaR и FavoWeb. [24]

Программный продукт FavoWeb – система оповещения об отказах, анализе и корректирующих действиях. [25]

FavoWeb использует возможности современных интернет-технологий и реализует полный замкнутый цикл методологии FRACAS, который применим к любому продукту, услуге и процессу. Применим в любой отрасли и фазе жизненного цикла изделий. Программа предлагает большой набор функций, возможностей оценки и улучшения надежности оборудования благодаря тесной интеграции с системой анализа RAM Commander. Базовая конфигурация программы включает возможности создания отчетов, проведения корректирующих действий, построения дерева продукта, работы с разными справочниками и библиотеками, кроме того, имеет модуль администрирования. [25]

Среди отечественных программных комплексов можно выделить:

* ПК АСОНИКА-К;
* ПК АСМ.

В ПК АСМ применяется общий логико-вероятностный метод системного анализа. Структура схема представляется в виде схемы функциональной целостности, которая позволяет отображать практически все известные виды структурных моделей систем. Комплекс автоматически формирует расчетные аналитические модели надежности и безопасности систем и вычисляет вероятность безотказной работы, среднюю наработку до отказа, коэффициент готовности, среднюю наработку на отказ и среднее время восстановления. [24]

АСОНИКА-К позволяет решать задачи анализа и повышения надежности радиоэлектронной аппаратуры, производимой как за рубежом, так и в России. Для анализа надежности аппаратуры применяются данные из справочников «Надежность электрорадиоизделий» и «Надежность электрорадиоизделий зарубежных аналогов». [25]

Состав программного комплекса АСОНИКА-К:

* система расчета характеристик надежности составных частей;
* система расчета показателей надежности изделий;
* система анализа результатов;
* система архивации проектов;
* справочная система;
* система сопровождения базы данных;
* система администрирования пользователей;
* система анализа и учета влияния на надежность внешних факторов;
* информационно-справочная система по характеристикам надежности; компонентов современной сложно-вычислительной техники и ЭРИ.

Использование отечественных программных комплексов более предпочтительно, так как они не уступают зарубежным аналогам и позволяют рассчитывать надежность аппаратуры отечественного производства.

**1.4 Цель и задачи автоматизации проектной процедуры прогнозирования надежности сложных технических систем.**

*Цель работы:* повышение качества и эффективности выполнения прогнозирования надежности сложных технических систем.

*Задачи работы:*

1. Провести анализ предметной области прогнозирования надежности сложных технических систем.
2. Сформулировать содержательную и математическую постановку задачи прогнозирования надежности сложных технических систем.
3. Разработать алгоритм прогнозирования надежности сложных технических систем.
4. Разработать информационное обеспечение автоматизированного прогнозирования надежности сложных технических систем.
5. Разработать программное обеспечение автоматизированного прогнозирования надежности сложных технических систем.

**Выводы по главе 1**

В рамках анализа проектной процедуры прогнозирования надежности сложных технических систем были достигнуты следующие результаты:

1. Разработаны функциональные и поведенческие модели проектной процедуры в виде IDEF0, IDEF3.
2. На основе диаграммы IDEF3 в дальнейшем будет разработана подсистема автоматизированного прогнозирования надежности сложных технических систем.
3. Обозначены цели и задачи прогнозирования надежности сложных технических систем.

**2. РАЗРАБОТКА МАТЕМАТИЧЕСКОГО ОБЕСПЕЧЕНИЯ ДЛЯ АВТОМАТИЗАЦИИ ПРОЕКТНОЙ ПРОЦЕДУРЫ ПРОГНОЗИРОВАНИЯ НАДЕЖНОСТИ СЛОЖНЫХ ТЕХНИЧЕСКИХ СИСТЕМ**

**2.1. Содержательная и математическая постановка задачи прогнозирования надежности сложных технических систем**

*Электронное средство* представляет собой сложную техническую систему, в основе функционирования которой лежат процессы передачи, извлечения и обработки информации, связанные с преобразованием и передачей электромагнитной энергии. [29]

Одним из самых важных свойств технических систем, ЭС в том числе, является *надежность*. [22]

Под сложной технической системой с точки зрения анализа надежности будем понимать ЭС, состоящее из произвольного количества произвольно соединенных звеньев (электронные модули, электронные устройства) с нагруженным резервированием (последовательно-параллельных и мостиковых структур).

Задачей расчета надежности ЭС является определение количественных показателей надежности.

В рамках данной выпускной квалификационной работы рассматриваются лишь невосстанавливаемые объекты.

*Исходные данные:*

* блок-схема надежности;
* перечень элементов;
* параметры элементов;
* планируемое время работы системы;

*Результирующие данные:*

* вероятность безотказной работы - вероятность того, что за указанное время работы отказ устройства не произойдет; [7]
* средняя наработка до отказа – среднее время, в течение которого устройство будет исправно работать (до возникновения первого отказа);
* гамма-процентная наработка до отказа - наработка, в течение которой отказ устройства не возникнет с заданной вероятностью; [7]
* интенсивность отказов - условная плотность вероятности возникновения отказа объекта, определяемая при условии, что до рассматриваемого момента времени отказ не возник. Численно равна числу отказов в единицу времени, отнесенное к числу узлов, безотказно проработавших до этого времени; [7]

Описание элементов математической модели приведено в таблице 1.

Таблица 1. Описание элементов математической модели

|  |  |  |
| --- | --- | --- |
| **Обозначение элемента мат. модели** | **Описание элемента мат. модели** | **Описание соответствующего компонента объекта проектирования** |
|  | Скалярная величина | Средняя наработка системы до отказа |
|  | Скалярная величина, | Вероятность безотказной работы системы в течение времени t |
|  | Скалярная величина, | Вероятность безотказной работы *i*-го элемента системы |
|  | Скалярная величина | Интенсивность отказов всей системы |
| *λ i* | Скалярная величина | Интенсивность отказов *i*-ого элемента системы |
| *Tγ* | Скалярная величина | Гамма-процентная наработка системы до отказа, |
|  | Скалярная величина | Количество элементов в системе |
|  | Скалярная величина, | Номер элемента |
| *t* | Скалярная величина | Время работы системы |

Используется предположение об экспоненциальном распределении времени до отказа элементов.

Рассмотрим простейшие типы структур – последовательное (см. рис. 6) и параллельное соединение (см. рис. 7) элементов.

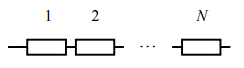


Рис. 6. Последовательное соединение элементов

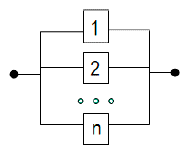


Рис. 7. Параллельное соединение элементов

Вероятность безотказной работы P(t) для последовательного и параллельного соединений вычисляется по формулам (1) и (2) соответственно

|  |  |
| --- | --- |
|  | (1) |

|  |  |
| --- | --- |
|  | (2) |

После определения значения вероятности безотказной работы системы, рассчитываются другие показатели надежности. По формулам (6), (7) и (8) вычисляется интенсивность отказов, средняя наработка до отказа и гамма-процентная наработка до отказа соответственно

|  |  |
| --- | --- |
|  | (3) |
|  |  |
|  | (4) |
|  |  |

|  |  |
| --- | --- |
|  | (5) |

Если же в качестве исходных данных выступает интенсивность отказов элементов системы, то сначала вычисляется вероятность безотказной работы каждого из них по формуле (6)

|  |  |
| --- | --- |
|  | (6) |

Реальные высоконадежные системы обычно представляют собой совокупность произвольно соединенных резервированных схем. Такие системы называют сложными последовательно-параллельными системами (ППС). ППС содержит ветви с параллельно соединенными элементами расчета надежности, совместный отказ которых приводит к отказу всей системы и ветви с последовательно соединенными элементами, отказ каждого из которых приводит к отказу системы. Пример подобных структурных схем надежности приведен на рис. 8.

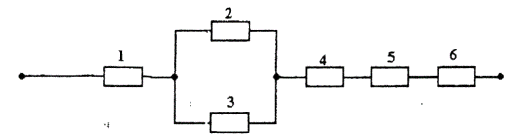


Рис. 8. Пример структурной схема надежности последовательно-параллельных систем

**2.2. Обзор методов и алгоритмов решения задачи прогнозирования надежности сложных технических систем**

На рис. 9 приведена классификация методов расчета надежности.

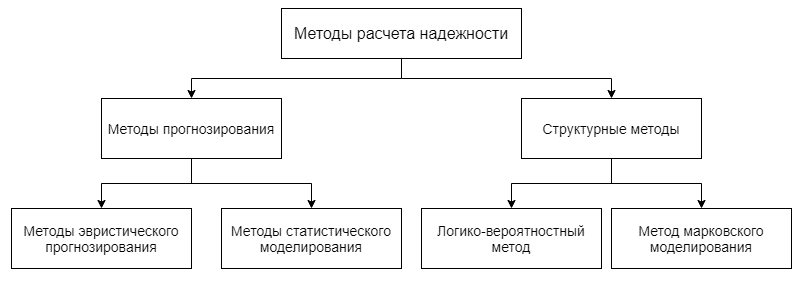


Рис. 9. Классификация методов расчета надежности

Наибольшее распространение получили *структурные методы* расчета надежности. Расчет показателей надежности структурными методами включает:

* представление объекта в виде структурной схемы;
* описание схемы с помощью адекватной математической модели.

В качестве структурных схем могут рассматриваться:

* блок-схемы надежности;
* деревья отказов;
* графы состояний.

Расчет надежности в рамках структурных методов чаще всего производится с помощью блок-схем надежности. Такая модель представляет взаимосвязи между компонентами с точки зрения надежности и не всегда соответствует реальному соединению элементов. [4]

*Логико-вероятностный метод* – метод анализа надежности, при котором структура системы описывается средствами математического аппарата бинарной алгебры логики, а количественная оценка надежности производится с помощью теории вероятностей. [4]

Расчет значений показателей надежности в момент времени t для системы, которая состоит из n элементов, осуществляется в три этапа:

1. формирование логической функции работоспособности системы;
2. преобразование логической функции к форме перехода к замещению;
3. получение расчетной вероятностной формулы.

Процессы, протекающие в системах с экспоненциальным распределением интервалов времени, являются *марковскими*, т.е. при которых вероятность перехода системы в новое состояние зависит только от состояния системы в настоящий момент и не зависит от того, когда и каким образом система перешла в это состояние. [15]

Построение марковских моделей надежности происходит следующим образом. На основе информации о структуре и принципах функционирования исследуемой системы определяется множество ее возможных состояний, которое разделяется на два подмножества: работоспособных состояний и состояний отказа. Строится граф переходов, вершинами которого являются состояния системы, а ребрами - возможные переходы между состояниями. Интенсивности переходов определяются характеристиками безотказности и ремонтопригодности элементов системы. По графу переходов составляется система уравнений, решение которой позволяет получить требуемые показатели надежности. Оценка параметров надежности технических систем с использованием графов позволяет учитывать любые факторы, влияющие на систему. [15]

В рамках *методов прогнозирования* надежности применяют имитационное статистическое моделирование и методы эвристического прогнозирования.

Имитационное моделирование (метод Монте-Карло) - численный метод проведения на ЭВМ вычислительных экспериментов с математическими моделями, имитирующими поведение реальных объектов и процессов. [12]

Методика статистического моделирования состоит из трех этапов:

1. моделирование на ЭВМ псевдослучайных последовательностей с заданным законом распределения вероятностей, имитирующих случайные значения параметров при каждом испытании;
2. использование полученных числовых последовательностей в имитационных математических моделях;
3. статистическая обработка результатов моделирования.

*Методы эвристического прогнозирования* основаны на статистической обработке независимых оценок значений ожидаемых показателей надежности разрабатываемого объекта (индивидуальных прогнозов), даваемых группой квалифицированных специалистов (экспертов) на основе предоставленной им информации об объекте, условиях его эксплуатации, планируемой технологии изготовления и других данных, имеющихся в момент проведения оценки. [19]

Результаты анализа методов сведены в таблицу 2.

Таблица 2. Результаты анализа методов

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Наименование алгоритма** | **Достоинства** | **Недостатки** |
| 1 | Логико-вероятностный алгоритм | 1) Применим при любой логической структуре схемы (не только при последовательно-параллельных схемах);  2) применим при любых распределениях наработки до отказа;  3) простая, относительно других структурных методов, реализация на ЭВМ. | 1) Не всегда удается составить логическую функцию работоспособности, достаточно точно соответствующую рассматриваемой системе;  2) для сложных систем c большим числом элементов преобразования с помощью функций алгебры логики становятся очень громоздкими;  3) применяется только для систем нагруженным резервированием. |
| 2 | Метод марковского моделирования | 1) Точное описание всех процессов, протекающих в сложных системах;  2) Удобное описание состояний системы с использованием графа переходов. | 1) При большом количестве элементов в системе, определение вероятностей безотказной работы связано с громоздкими вычислениями;  2) с ростом числа элементов трудности моделирования надежности сложных технических систем быстро возрастают из-за большого количества возможных состояний;  3) сложная программная реализация, требующая применения аналитических методов решения уравнений. |

Таблица 2 (продолжение)

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Наименование алгоритма** | **Достоинства** | **Недостатки** |
| 3 | Метод прогнозирования (метод Монте-Карло) | 1) Простая реализация на ЭВМ;  2) позволяет имитировать реальные физические процессы. | 1) Высокая достоверность имитационного моделирования практически недостижима;  2) при анализе сложных систем возникают проблемы, связанные с большими затратами машинного времени, необходимого для вычислений. |
| 4 | Метод эвристического прогнозирования | 1) Объединении групп элементов системы в один эквивалентный элемент, тем самым происходит уменьшение числа элементов в системе;  2) простая реализация на ЭВМ. | 1) Не позволяет установить погрешность вычислений;  2) применяется исключительно для случая высоконадежных элементов и систем. |

**2.3. Разработка алгоритма решения задачи прогнозирования надежности сложных технических систем**

Для решения задачи прогнозирования надежности электронного средства будем использовать логико-вероятностный метод (ЛВМ). Причины выбора данного метода – удобное описание логической функции системы любой структуры с использованием средств математического аппарата бинарной алгебры логики и простая, в сравнении с другими структурными методами, программная реализация.

Основная идея логико-вероятностного метода заключается в описании структуры системы средствами математического аппарата бинарной алгебры логики. Количественная оценка надежности производится с помощью теории вероятностей.

Блок-схема алгоритма приведена на рисунке 10.

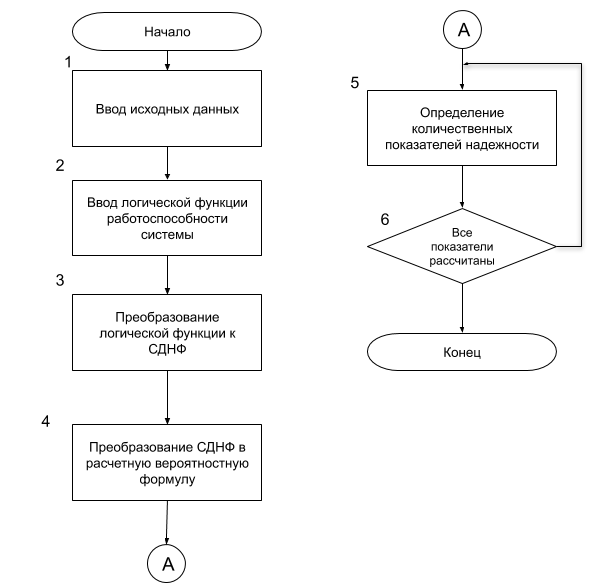


Рис. 10. Блок-схема алгоритма

В качестве исходных данных выступает вероятность безотказной работы элементов (блоки, узлы, электронные устройства) и логическая функция работоспособности, составленная по структурной схеме надежности (блок-схема надежности).

На основе анализа блок-схемы надежности, составляется логическая функция работоспособности системы [4]. Предполагая, что система и ее составляющие компоненты могут находиться только в двух состояниях – работоспособности и отказа, а отказы компонентов не зависят друг от друга, можно сформировать логическую функцию ее работоспособности S(x) с помощью формулы (9)

|  |  |
| --- | --- |
|  | (7) |

Аргументом функции S является вектор-строка *x* логических переменных, которая определяется формулой (10)

|  |  |
| --- | --- |
|  | (8) |

Например, если система состоит из двух элементов и отказ каждого приводит к отказу всей системы, т.е. элементы с точки зрения надежности соединены последовательно (см. рис. 6), то. Функция работоспособности параллельной схемы (см. рис. 7), состоящей из 2 элементов, отказы одиночные отказы элементов которой не приводят к ее отказу, равна [4]

После составления логической функции происходит ее преобразование к форме перехода к замещению. Наиболее удобной формой представления, в плане простоты реализации на ЭВМ, является совершенная дизъюнктивная нормальная форма (СДНФ). Если использование ЭВМ для реализации ЛВМ не планируется, то рекомендуется выбрать иную форму перехода к замещению, так как СДНФ является достаточной громоздкой.

После преобразования исходной логической функции работоспособности в СДНФ, получение расчетной формулы для вероятности истинности в момент времени t логической функции системы не представляет труда:

* логическую переменную «xi»необходимо заменить на соответствующую вероятность «pi = P(xi)»;
* логическую операцию дизъюнкции необходимо заменить на операцию сложения «+»;
* логическую операцию конъюнкции необходимо заменить на операцию умножения «\*»;
* логическое отрицание «» заменяется на «1-P(xi)».

После расчета по полученной формуле получаем значение вероятности безотказной работы. Остальные показатели надежности определяются по формулам (3) – (5).

На простом примере разберем, как осуществляется расчет надежности с помощью логико-вероятностного метода.

Исходные данные: перечень элементов, интенсивности отказов, блок-схема надежности системы, время работы ч, гамма процент = 95%.

Блок-схема надежности рассматриваемой системы представлена на рис. 11.

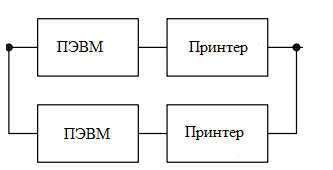


Рис. 11. Блока-схема надежности устройства

Структура представляет собой общее резервирование из двух каналов и 4 элементов.

Исходные данные занесены в таблицу 3.

Таблица 3. Исходные данные

|  |  |  |  |
| --- | --- | --- | --- |
| **Номер элемента** | **Наименование**  **элемента системы** |  | **Логическая переменная** |
| 1 | ПЭВМ | 1 | A |
| 2 | C |
| 3 | Принтер | 5 | B |
| 4 | D |

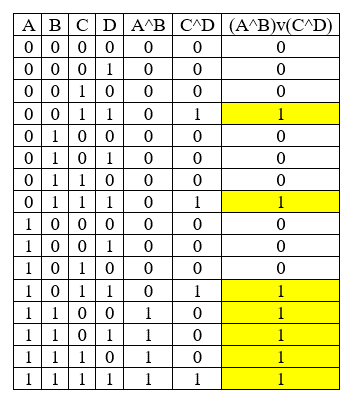
По формуле (6) определим вероятности безотказной работы элементов:

Пользуясь формулами (7) и (8), составим логическую функцию работоспособности системы:

Преобразуем исходную логическую функцию в совершенную дизъюнктивную нормальную форму. Для этого составим таблицу истинности.

Таблица истинности для логической функции работоспособности системы представлена в таблице 4.

Таблица 4. Таблица истинности



Для нахождения СДНФ нужно из таблицы истинности выделить лишь те строки, результат которых равен 1. Далее, для каждой строки выписываем конъюнкцию всех переменных по следующему алгоритму: если значение переменной в данной строке равно 1, то в конъюнкцию записываем саму переменную, а если равно 0, то - отрицание этой переменной. После этого все конъюнкции связываем в дизъюнкцию.

В результате, совершенная дизъюнктивно-нормальная форма (СДНФ) нашей функции равна:

Заменяя логические операции на алгебраические, а логические переменные на соответствующие вероятности безотказной работы, получаем расчетную формулу вероятности безотказной работы системы:

Рассчитываем остальные показатели надежности по формулам (3), (4) и (5).

**Выводы по главе 2**

1. Сформулирована содержательная и математическая постановки задачи прогнозирования надежности сложных технических систем.
2. Проведен анализ методов решения задачи прогнозирования надежности сложных технических систем.
3. Выбран логико-вероятностный метод для прогнозирования надежности сложных технических систем.
4. Рассмотрено решение задачи прогнозирования надежности сложных технических систем с помощью логико-вероятностного метода на контрольном примере.

**3. РАЗРАБОТКА ИНФОРМАЦИОННОГО ОБЕСПЕЧЕНИЯ АВТОМАТИЗИРОВАННОЙ ПРОЕКТНОЙ ПРОЦЕДУРЫ ПРОГНОЗИРОВАНИЯ НАДЕЖНОСТИ СЛОЖНЫХ ТЕХНИЧЕСКИХ СИСТЕМ**

**3.1. Разработка модели потока данных (DFD) и концептуальной модели базы данных**

Модель окружения представлена на рис. 12.

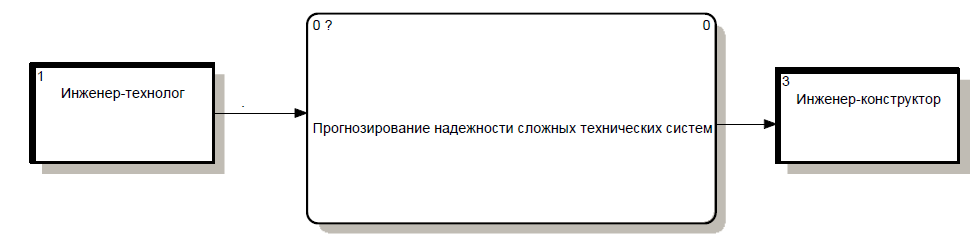


Рис. 12. Модель окружения

Инженер-технолог выполняет автоматизированный расчет надежности системы. После чего он передает результаты расчета инженеру-конструктору, чтобы тот сформировал отчет о надежности технической системы.

Логическая модель проектной процедуры представлена на рис. 13.

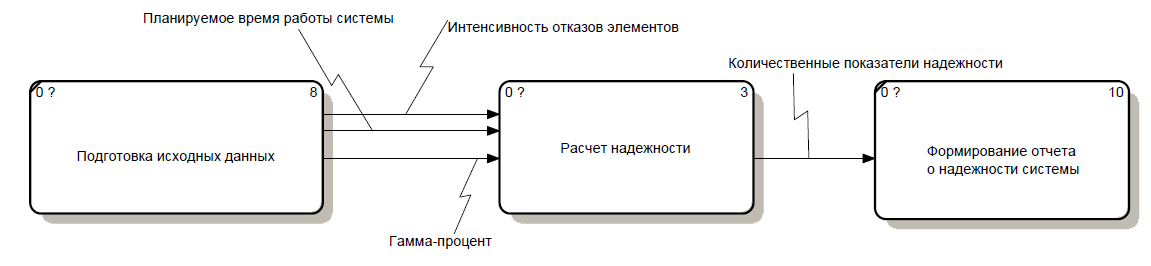


Рис. 13. Логическая модель проектной процедуры

Происходит ввод исходных данных: интенсивность отказов элементов, планируемое время работы системы и гамма-процент. Далее осуществляется расчет надежности с использованием логико-вероятностного метода: на основе анализа структурной-схемы надежности, составляется логическая функция работоспособности системы, логическая функция работоспособности системы преобразуется в совершенную дизъюнктивную нормальную форму, а СДНФ – в расчетную формулу. Результатом расчета являются количественные показатели надежности: вероятность безотказной работы, средняя наработка до отказа, интенсивность отказов и гамма-процентная наработка до отказа. В конечном счете, рассчитанные показатели надежности используются для оформления отчета о надежности технической системы.

Модель поведения представлена на рис. 14.

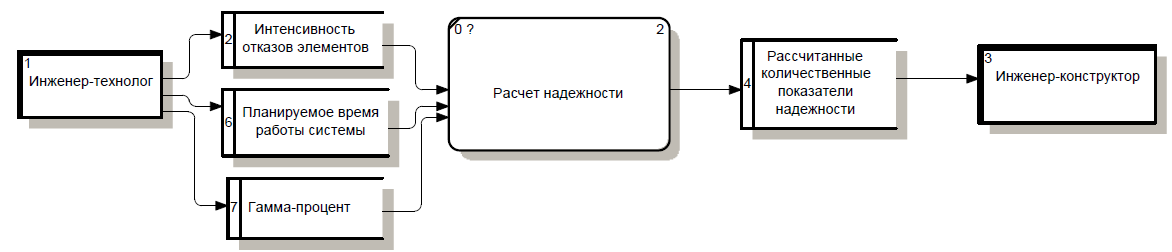


Рис. 14. Модель поведения

Инженер технолог вводит все необходимые данные для расчета надежности в программу: интенсивность отказов элементов, планируемое время работы системы и гамма-процент. Затем осуществляется автоматизированный расчет надежности системы. Результатом расчета являются количественные показатели надежности, которые передаются инженеру-конструктору. Он проводит анализ полученных результатов. Если надежность удовлетворяет требованиям ТЗ, инженер-конструктор составляет отчет о надежности технической системы. В противном случае, в отчет о надежности технической системы, помимо самих показателей надежности, инженер-конструктор также записывает рекомендации по повышению надежности технической системы.

Начнем формирование базы данных (БД) с установления состава сущностей. Название сущностей, их описание и количество экземпляров представлено в таблице 5.

Таблица 5. Сущности

|  |  |  |
| --- | --- | --- |
| **Обозначение сущности** | **Наименование сущности** | **Количество экземпляров** |
|  | Класс | 15 |
|  | Электрорадиоизделие | 100 |

Сущность «Класс» необходима для хранения списка классов ЭРИ.

Сущность «Электрорадиоизделие» содержит список всех ЭРИ и интенсивность отказов к каждому изделию.

Для сущности «Класс» атрибутами будут: «Идентификационный номер класса» (первичный ключ), «Наименование класса» («Оптоэлектронные приборы», «Интегральные микросхемы» и т.п.).

Для сущности «Электрорадиоизделие» атрибутами будут: «Идентификационный номер ЭРИ» (первичный ключ), «Наименование ЭРИ», «Интенсивность отказов», «Идентификационный номер класса» (внешний ключ к сущности «Класс»).

Атрибуты и домены для каждой сущности представлены в таблице 6.

Таблица 6. Описание атрибутов

|  |  |  |
| --- | --- | --- |
| **Наименование сущности** | **Наименование атрибута** | **Наименование домена** |
| Класс | Идентификационный номер класса | Номер |
| Наименование класса | Название |
| Электрорадиоизделие | Идентификационный номер ЭРИ | Номер |
| Наименование ЭРИ | Название |
| Интенсивность отказов | Числовое значение |
| Идентификационный номер класса | Номер |

Описание доменов представлено в таблице 7.

Таблица 7. Описание доменов

|  |  |  |
| --- | --- | --- |
| **Наименование домена** | **Тип данных** | **Примеры значений** |
| Номер | Целочисленный тип данных | 1,2,3…999 |
| Название | Строковой тип данных | Конденсатор |
| Числовое значение | Вещественный тип данных | 0,5;10,3;99,7 |

Составим концептуальную модель для базы данных. Ее графическая модель (ER-диаграмма) представлена на рис. 15.

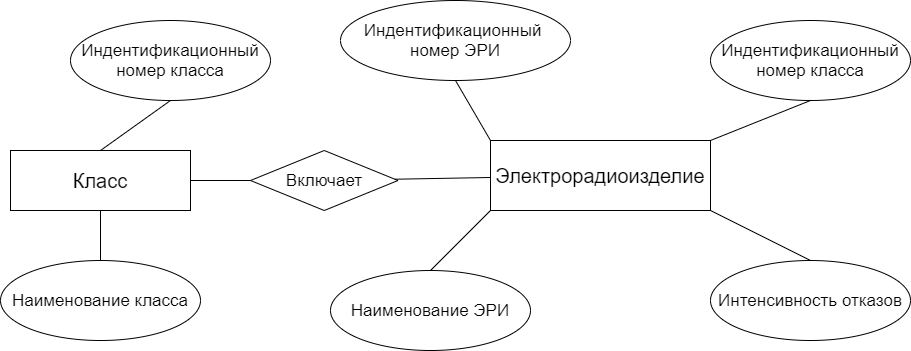


Рис. 15. ER-диаграмма

**3.2. Логическое проектирование базы данных**

Описание логического проектирования и связь атрибутов представлены в таблице 8.

Таблица 8. Связь атрибутов

|  |  |  |
| --- | --- | --- |
| **Наименование**  **сущности** | **Наименование атрибута** |  |
| Класс | Идентификационный номер класса |  |
| Наименование класса |
| Электрорадиоизделие | Идентификационный номер ЭРИ |  |
| Наименование ЭРИ |
| Интенсивность отказов |
| Идентификационный номер класса |

Все отношения находятся в третьей нормальной форме. Диаграмма логической модели представлена на рис. 16.

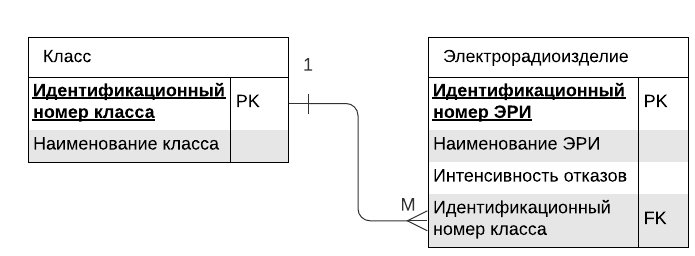


Рис. 16. Логическая модель базы данных

**3.3. Физическое проектирование базы данных**

Реализуем базу данных в Microsoft SQL Server (MSS). Управление компонентами MSS будем осуществлять с помощью утилиты Microsoft SQL Server Management Studio.

Рассмотрим содержание всех таблиц.

На рис. 17 представлена таблица сущности «Класс».

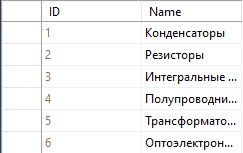


Рис. 17. Таблица сущности «Класс»

На рис. 18 представлена таблица сущности «Электрорадиоизделие».

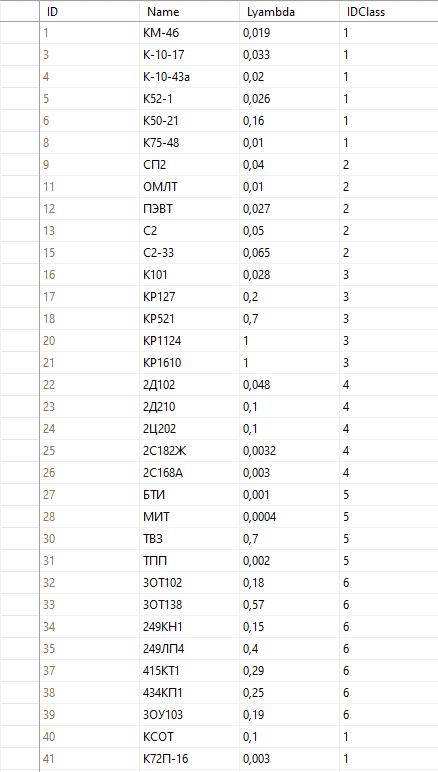


Рис. 18. Таблица сущности «Электрорадиоизделие»

**Выводы по главе 3**

1. Были установлены сущности и определены совокупности атрибутов для каждой сущности; определены первичные и внешние ключи для каждой сущности; выявлены связи между сущностями и построена концептуальная модель базы данных.
2. Была построена логическая модель базы данных.
3. База данных была реализована на ЭВМ в Microsoft SQL Server.

**4. РАЗРАБОТКА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ АВТОМАТИЗАЦИИ ПРОЕКТНОЙ ПРОЦЕДУРЫ** **ПРОГНОЗИРОВАНИЯ НАДЕЖНОСТИ СЛОЖНЫХ ТЕХНИЧЕСКИХ СИСТЕМ**

**4.1. Архитектура программы для автоматизации проектной процедуры прогнозирования надежности сложных технических систем**

**4.1.1. Анализ функциональных требований к программе**

Диаграмма вариантов использования представлена на рис. 19.

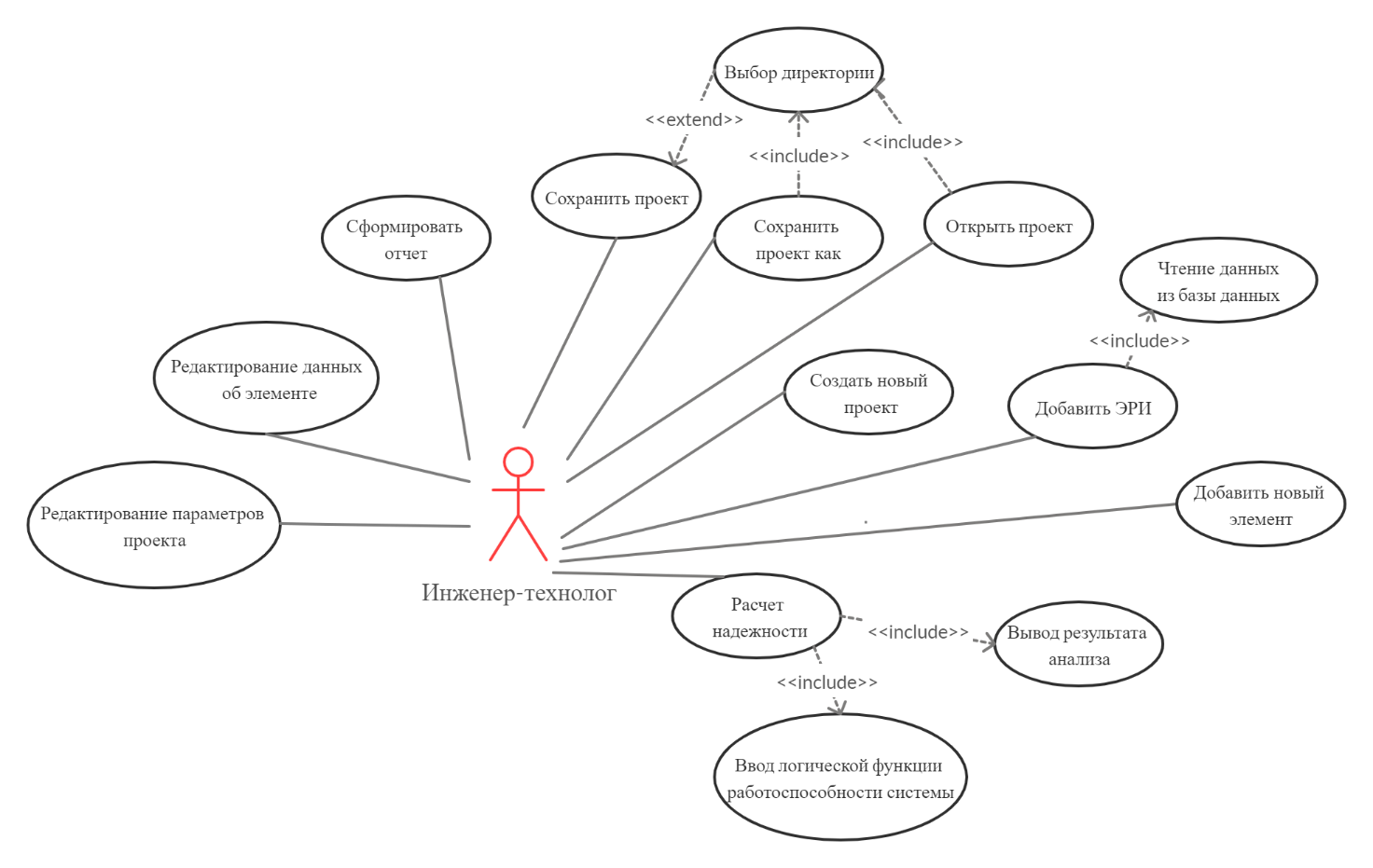
****

Рис. 19. Диаграмма вариантов использования

Сценарий №1 «Создать новый проект». Применяется для создания нового проекта. Инженер-технолог выбирает опцию меню «Создать проект». Программа открывает диалоговое окно для того, чтобы пользователь ввел наименование проекта. Программа проверяет корректность введенных данных и закрывает диалоговое окно.

Вариант использования: создать новый проект.

Актеры: инженер-технолог.

Цель: создать новый проект.

Тип: базовый.

Сценарий для варианта использования «Создать проект» представлен в таблице 9.

Таблица 9. Создать новый проект

|  |  |
| --- | --- |
| **Инженер-технолог** | **Программа** |
| 1. Нажимает на кнопку «Проект», затем на «Создать новый» | 2. Открывает диалоговое окно и предлагает ввести наименование проекта |
| 3. Вводит наименование проекта | 4. Проверяет корректность введенных данных |
| Исключение 1: «Входная строка с наименованием проекта была пустой» | |
|  | 5. Закрывает диалоговое окно и показывает пользователю пустой список элементов, входящих в техническую систему |

Сценарий №2 «Сохранить проект». Применяется для сохранения проекта. Проект сохраняется в отдельный файл. Инженер-технолог выбирает опцию меню «Сохранить проект». Программа проверяет, сохранялся ли данный проект ранее. Если сохранялся, программа перезаписывает файл. В противном случае программа открывает диалоговое окно, в котором пользователю необходимо ввести имя файла и указать директорию, в которую он хочет сохранить файл.

Вариант использования: сохранить проект.

Актеры: инженер-технолог.

Цель: сохранить проект.

Тип: базовый.

Сценарий для варианта использования «Сохранить проект» представлен в таблице 10.

Таблица 10. Сохранить проект

|  |  |
| --- | --- |
| **Инженер-технолог** | **Программа** |
| 1. Нажимает на кнопку «Проект», затем на «Сохранить» | 2. Проверяет, сохранялся ли проект ранее |
| Исключение 2: «Проект еще ни разу не сохранялся» | |
|  | 3. Перезаписывает ранее сохраненный файл. |

Сценарий №3 «Сохранить проект как». Применяется для сохранения проекта по указанному пути. Инженер-технолог выбирает опцию меню «Сохранить проект как». Программа открывает диалоговое окно, в котором пользователю необходимо ввести имя файла и указать директорию, в которую он хочет сохранить файл.

Вариант использования: сохранить проект как.

Актеры: инженер-технолог.

Цель: сохранить проект в указанной директории.

Тип: базовый.

Сценарий для варианта использования «Сохранить проект как» представлен в таблице 11.

Таблица 11. Сохранить проект как

|  |  |
| --- | --- |
| **Инженер-технолог** | **Программа** |
| 1. Нажимает на кнопку «Проект», затем на «Сохранить как» | 2. Показывает обозреватель файловой системы. Предлагает выбрать место сохранения проекта. |
| 3. Вводит имя проекта, указывает директорию и нажимает на «Сохранить» | 4. Сохраняет проект в указанной директории. |

Сценарий №4 «Открыть проект». Применяется для открытия проекта. Инженер-технолог нажимает на опцию меню «Открыть проект». Программа открывает обозреватель файловой системы. Пользователь выбирает файл с проектом. Программа проверяет файл. Если файл поврежден, то программа предлагает пользователю создать новый проект.

Вариант использования: открыть проект.

Актеры: инженер-технолог.

Цель: открыть существующий проект.

Тип: базовый.

Сценарий для варианта использования «Открыть проект» представлен в таблице 12.

Таблица 12. Открыть проект

|  |  |
| --- | --- |
| **Инженер-технолог** | **Программа** |
| 1. Нажимает на кнопку «Проект», затем на «Открыть» | 2. Показывает обозреватель файловой системы. Предлагает выбрать проект. |
| 3. Выбирает существующий проект | 4. Проверяет целостность файла и пытается его открыть |
| Исключение 3: «Ошибка открытия проекта» | |
|  | 5. Открывает проект, выбранный пользователем. |

Сценарий №5 «Добавить новый элемент». Применяется для добавления нового элемента в техническую систему. Инженер-технолог нажимает на правую кнопку мыши по дереву проекта, затем выбирает в выпадающем меню «Добавить компонент-Новый элемент». Программа открывает диалоговое окно, в котором пользователю необходимо ввести наименования элемента. Пользователь вводит наименование элемента, после чего программа проверяет корректность ввода. Если строка с названием элемента не была пустой, программа добавляет элемент в список элементов технической системы.

Вариант использования: добавить новый элемент.

Актеры: инженер-технолог.

Цель: добавить новый элемент в техническую систему.

Тип: базовый.

Сценарий для варианта использования «Добавить новый элемент» представлен в таблице 13.

Таблица 13. Добавить новый элемент

|  |  |
| --- | --- |
| **Инженер-технолог** | **Программа** |
| 1. Нажимает на правую кнопку мыши по дереву проекта, затем выбирает в выпадающем меню «Добавить компонент-Новый элемент» | 2. Открывает диалоговое меню и предлагает пользователю ввести наименование элемента |
| 3. Вводит наименование проекта | 4. Проверяет корректность введенных данных |
| Исключение 4: «Входная строка с наименованием элемента была пустой» | |
|  | 5. Добавляет элемент в список элементов системы. |

Сценарий №6 «Добавить ЭРИ». Применяется для добавления ЭРИ из базы данных в техническую систему. Инженер-технолог нажимает на правую кнопку мыши по дереву проекта, затем выбирает в выпадающем меню «Добавить компонент-ЭРИ из базы данных». Программа подключается к базе данных и открывает окно с выбором класса ЭРИ и самого элемента. Пользователь осуществляет выбор, после чего программа закрывает окно.

Вариант использования: добавить ЭРИ.

Актеры: инженер-технолог.

Цель: добавить ЭРИ в список элементов.

Тип: базовый.

Сценарий для варианта использования «Добавить ЭРИ» представлен в таблице 14.

Таблица 14. Добавить ЭРИ

|  |  |
| --- | --- |
| **Инженер-технолог** | **Программа** |
| 1. Нажимает на правую кнопку мыши по дереву проекта, затем выбирает в выпадающем меню «Добавить компонент-ЭРИ» | 2. Подключается к базе данных |
| Исключение 5: «Ошибка при подключении к базе данных» | |
|  | 3. Открывает окно с выбором класса ЭРИ и самого элемента |

Таблица 14 (продолжение)

|  |  |
| --- | --- |
| **Инженер-технолог** | **Программа** |
| 4. Осуществляет выбор | 5. Закрывает окно с выбором и добавляет выбранный элемент в список элементов. |

Сценарий №7 «Редактирование параметров проекта». Применяется для редактирования параметров проекта. Инженер-технолог нажимает левой кнопкой мыши по названию проекта в дереве проекта. Программа открывает вкладку, где пользователь может изменить имя проекта, время планируемой работы технической системы и гамма-процент.

Вариант использования: редактирование параметров проекта.

Актеры: инженер-технолог.

Цель: Отредактировать параметры проекта.

Тип: базовый.

Сценарий для варианта использования «Редактирование параметров проекта» представлен в таблице 15.

Таблица 15. Редактирование параметров проекта

|  |  |
| --- | --- |
| **Инженер-технолог** | **Программа** |
| 1. Нажимает левой кнопкой мыши по названию проекта в дереве проекта | 2. Открывает вкладку для редактирования параметров проекта |
| 3. По желанию редактирует любой из параметров: наименование проекта, время планируемой работы технической системы и гамма-процент | 4. Проверяет корректность введенных данных |
| Исключение 6: «Неверный формат данных» | |
|  | 5. Сохраняет измененные параметры. |

Сценарий №8 «Редактирование данных об элементе». Применяется для редактирования данных об элементе. Инженер-технолог нажимает левой кнопкой мыши по элементу в дереве проекта. Программа открывает вкладку, где пользователь может изменить наименование элемента и его интенсивность отказов.

Вариант использования: редактирование данных об элементе.

Актеры: инженер-технолог.

Цель: отредактировать параметры элемента.

Тип: базовый.

Сценарий для варианта использования «Редактирование данных об элементе» представлен в таблице 16.

Таблица 16. Редактирование данных об элементе

|  |  |
| --- | --- |
| **Инженер-технолог** | **Программа** |
| 1. Выбирает элемент в дереве проекта, параметры которого он хочет отредактировать. | 2. Открывает вкладку для редактирования данных об элементе |
| 3. По желанию редактирует любой из параметров: наименование элемента и интенсивность отказов | 4. Проверяет корректность введенных данных |
| Исключение 6: «Неверный формат данных» | |
|  | 5. Сохраняет измененные данные. |

Сценарий №9 «Расчет надежности». Применяется для расчета надежности технической системы. Инженер-технолог нажимает на кнопку «Расчет надежности». Программа открывает специальную форму для ввода логической функции работоспособности системы. Пользователь вводит логическую функцию. Программа проверяет корректность данных, закрывает форму, осуществляет расчет надежности технической системы и вывод результата расчета на экран.

Вариант использования: расчет надежности.

Актеры: инженер-технолог.

Цель: осуществить расчет надежности технической системы.

Тип: базовый.

Сценарий для варианта использования «Расчет надежности» представлен в таблице 17.

Таблица 17. Расчет надежности

|  |  |  |
| --- | --- | --- |
| **Инженер-технолог** | **Программа** | |
| 1. Нажимает на кнопку «Расчет надежности» |  | |
| Исключение 7: «Список элементов пуст» | | |
|  | 2. Показывает форму для ввода логической функции работоспособности системы | |
| 3. Вводит логическую функцию | 4. Проверяет корректность введенных данных | |
| Исключение 8: «Логическая функция введена некорректно» | | |
|  | | 5. Закрывает форму и осуществляет расчет надежности технической системы |
|  | | 6. Добавляет на главную форму программы две новые вкладки – «Параметры безотказности», где пользователь может посмотреть на результаты расчета, и «Пошаговое решение», где будет расписано решение по шагам. |

Сценарий №10 «Сформировать отчет». Используется для формирования отчета с рассчитанными показателями надежности. Инженер-технолог нажимает на опцию меню «Сформировать отчет». Если расчет не проводился, то программа выводит соответствующее сообщение. Иначе формирует отчет в формате файлов Excel и сохраняет его в корневой папке программы.

Вариант использования: сформировать отчет.

Актеры: инженер-технолог.

Цель: сформировать отчет с рассчитанными показателями надежности системы.

Тип: базовый.

Сценарий для варианта использования «Сформировать отчет» представлен в таблице 18.

Таблица 18. Сформировать отчет

|  |  |  |
| --- | --- | --- |
| **Инженер-технолог** | | **Программа** |
| 1. Нажимает на кнопку «Сформировать отчет» | |  |
| Исключение 9: «Расчет надежности не проводился» | | |
|  | 2. Формирует отчет в формате «Таблица Excel (\*.xls)» и сохраняет его в корневой папке программы | |

Исключение «Входная строка с наименованием проекта была пустой» представлено в таблице 19.

Таблица 19. Исключение «Входная строка с наименованием проекта была пустой»

|  |  |
| --- | --- |
| **Инженер-технолог** | **Программа** |
| 3. Вновь вводит наименование проекта | 4. Проверяет корректность введенных данных |

Исключение «Проект еще ни разу не сохранялся» представлено в таблице 20.

Таблица 20. Исключение «Проект еще ни разу не сохранялся»

|  |  |
| --- | --- |
| **Инженер-технолог** | **Программа** |
|  | 3. Показывает обозреватель файловой системы. Предлагает выбрать место сохранения проекта. |
| 4. Вводит имя проекта и нажимает на «Сохранить» | 5. Сохраняет проект в указанной директории. |

Исключение «Ошибка открытия проекта» представлено в таблице 21.

Таблица 21. Исключение «Ошибка открытия проекта»

|  |  |
| --- | --- |
| **Инженер-технолог** | **Программа** |
|  | 5. Сообщает о повреждении файла проекта и предлагает создать новый проект. |

Исключение «Входная строка с наименованием элемента была пустой» представлено в таблице 22.

Таблица 22. Исключение «Входная строка с наименованием элемента была пустой»

|  |  |
| --- | --- |
| **Инженер-технолог** | **Программа** |
| 3. Вновь вводит наименование проекта | 4. Проверяет корректность введенных данных |

Исключение «Ошибка при подключении к базе данных» представлено в таблице 23.

Таблица 23. Исключение «Ошибка при подключении к базе данных»

|  |  |
| --- | --- |
| **Инженер-технолог** | **Программа** |
|  | 3. Сообщает об ошибке при подключении к базе данных |
| 4. Решает проблемы с подключением к базе данных |  |

Исключение «Неверный формат данных» представлено в таблице 24. В случае, когда происходит редактирование параметров проекта, данное исключение возникает, если пользователь оставил строку с наименованием проекта пустой или ввел не числовое значение в поля редактирования гамма-процента и времени планируемой работы технической системы. В случае с редактированием данных об элементе, данное исключение возникает, если пользователь оставил строку с наименованием элемента пустой или ввел не числовое значение в поле редактирования интенсивности отказов.

Таблица 24. Исключение «Неверный формат данных»

|  |  |
| --- | --- |
| **Инженер-технолог** | **Программа** |
| 3. Вновь редактирует любой из параметров. | 4. Проверяет корректность введенных данных |

Исключение «Список элементов пуст» представлено в таблице 25.

Таблица 25. Исключение «Список элементов пуст»

|  |  |
| --- | --- |
| **Инженер-технолог** | **Программа** |
|  | 5. Сообщает пользователю о том, что ему необходимо добавить в проект хотя бы один элемент |

Исключение «Логическая функция введена некорректно» представлено в таблице 26.

Таблица 26. Исключение «Логическая функция введена некорректно»

|  |  |
| --- | --- |
| **Инженер-технолог** | **Программа** |
| 3. Вновь вводит логическую функцию | 4. Проверяет корректность введенных данных |

Исключение «Расчет надежности не проводился» представлено в таблице 27.

Таблица 27. Исключение «Расчет надежности не проводился»

|  |  |
| --- | --- |
| **Инженер-технолог** | **Программа** |
|  | 2. Сообщает пользователю о том, что ему необходимо провести расчет надежности перед оформлением отчета |

**4.1.2. Описание архитектуры программы**

Для решения задачи прогнозирования надежности сложных технических систем была реализована программа с использованием логико-вероятностного метода.

Соединение с базой данных происходит через connectionString. После подключения программы к базе данных, Entity Framework производит автоматическую генерацию классов-сущностей. Таким образом, вместо оперирования таблицами, индексами, первичными и внешними ключами, работа с данными будет осуществляться на уровне объектов классов-сущностей.

Диаграмма классов представлена на рис. 20.

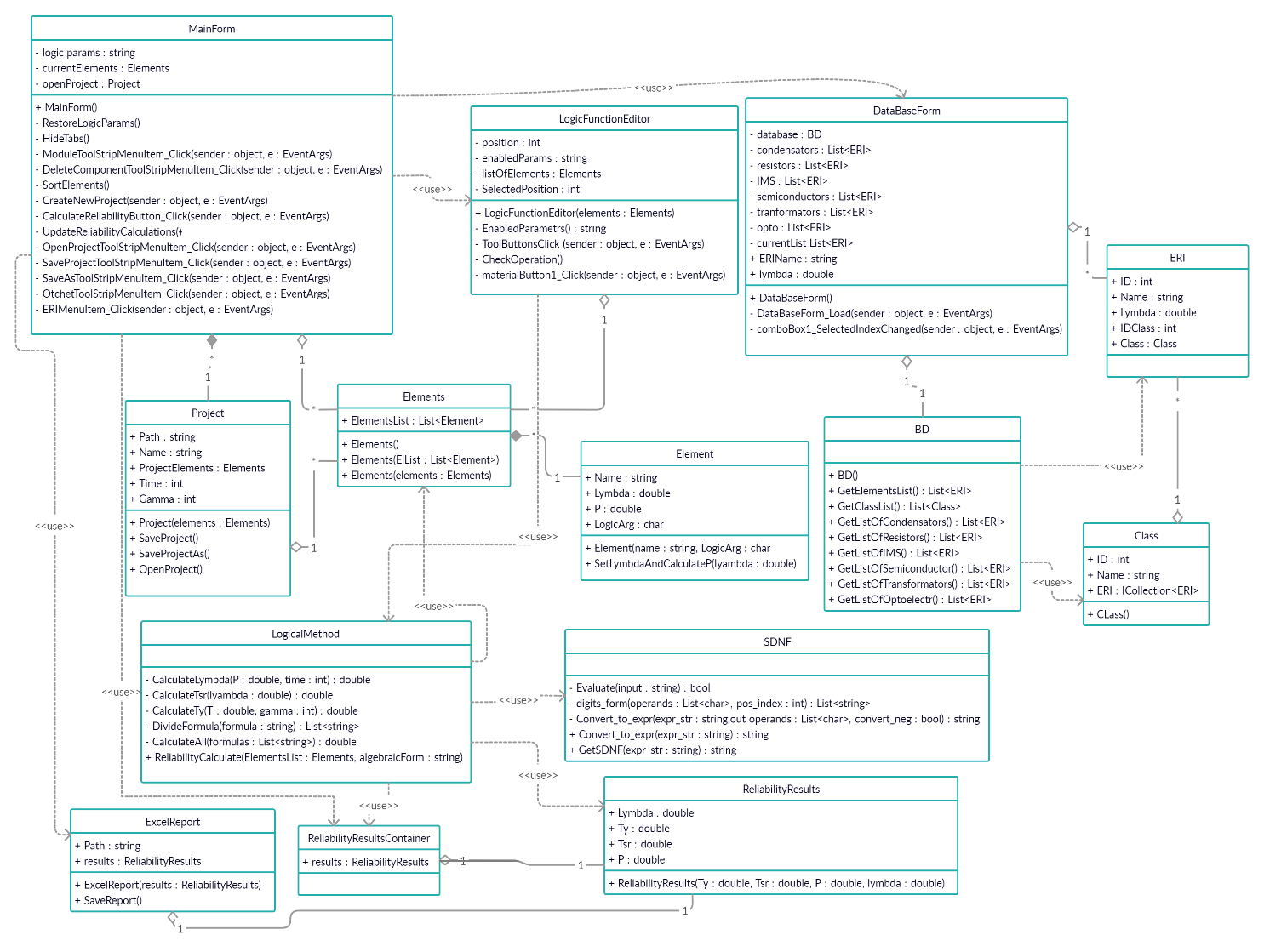


Рис. 20. Диаграмма классов

В таблице 28 представлены методы и атрибуты класса MainForm.

Таблица 28. Методы и атрибуты класса MainForm

|  |  |
| --- | --- |
| **Наименование** | **Описание** |
| string logicParametrs | Набор доступных логических переменных для элементов |
| Elements currentElements | Список элементов системы |
| Project openProject | Открытый проект (если пользователь воспользуется функцией «Проект-Открыть проект» |
| MainForm() | Конструктор класса |
| RestoreLogicParams() | Восстанавливает набор доступных логических переменных |
| HideTabs() | Скрывает вкладки с результатами расчета до тех пор, пока расчет не будет произведен |
| ModuleToolStripMenuItem\_Click(  object sender, EventArgs e) | Добавляет новый элемент в проект |
| DeleteToolStripMenuItem\_Click(  object sender, EventArgs e) | Удаляет элемент, выбранный пользователем |
| SortElements() | Сортирует элементы в алфавитном порядке по их логической переменной |
| CreateNewProject(object sender, EventArgs e) | Создает новый проект |
| CalculateReliabilityButton\_Click(  object sender, EventArgs e) | Запускает процесс расчета надежности |
| UpdateReliabilityCalculations() | Обновляет вкладку с результатами расчета надежности и вкладку решением по шагам |

Таблица 28. Продолжение

|  |  |
| --- | --- |
| **Наименование** | **Описание** |
| OpenProjectToolStripMenuItem\_Click(object sender, EventArgs e) | Запускает процесс открытия существующего проекта |
| SaveProjectToolStripMenuItem\_Click(object sender, EventArgs e) | Запускает процесс сохранения текущего проекта |
| SaveAsToolStripMenuItem\_Click(object sender, EventArgs e) | Запускает процесс сохранения текущего проекта в определенной директории |
| OtchetToolStripMenuItem\_Click(  object sender, EventArgs e) | Запускает процесс формирования отчета |
| ERIMenuItem\_Click(object sender, EventArgs e) | Запускает процесс добавления ЭРИ из базы данных в проект |

В таблице 29 представлены методы и атрибуты класса LogicFunctionEditor.

Таблица 29. Методы и атрибуты класса LogicFunctionEditor

|  |  |
| --- | --- |
| **Наименование** | **Описание** |
| int position | Позиция каретки ввода в поле ввода логической функции работоспособности системы |
| string enabledParams | Доступные логические переменные |
| Elements listOfElements | Список элементов системы |
| int selectedPosition | Свойство, описывающее атрибут position |
| LogicFunctionEditor(Elements elements) | Конструктор класса |

Таблица 29. Продолжение

|  |  |
| --- | --- |
| **Наименование** | **Описание** |
| string EnableParametrs() | Включает только те кнопки, логическая переменная которых входит в атрибут enabledParams |
| ToolButtonsClick(object sender, EventArgs e) | Событие, обрабатывающее нажатие на любую кнопку в калькуляторе |
| CheckOperations() | Проверяет корректность ввода логической функции |
| materialButton1\_Click(object sender, EventArgs e) | Запускает расчет надежности технической системы, на основе введенной логической функции, с применением логико-вероятностного метода |

В таблице 30 представлены методы и атрибуты класса DataBaseForm.

Таблица 30. Методы и атрибуты класса DataBaseForm

|  |  |
| --- | --- |
| **Наименование** | **Описание** |
| BD database | Объект для получения данных из БД |
| List<ERI> condensators | Список конденсаторов |
| List<ERI> resistors | Список резисторов |
| List<ERI> IMS | Список интегральных микросхем |
| List<ERI> semiconductors | Список полупроводниковых приборов |
| List<ERI> transformators | Список трансформаторов |
| List<ERI> opto | Список оптоэлектронных приборов |
| List<ERI> currentList | Список ЭРИ, составленный на основе выбора пользователя |

Таблица 30. Продолжение

|  |  |
| --- | --- |
| **Наименование** | **Описание** |
| string ERIName | Наименование выбранного ЭРИ |
| double lymbda | Интенсивность отказов выбранного ЭРИ |
| DataBaseForm() | Конструктор класса |
| DataBaseForm\_Load(object sender, EventArgs e) | Инициализирует все необходимые атрибуты при запуске формы |
| comboBox1\_SelectedIndexChanged(  object sender, EventArgs e) | Выводит список ЭРИ выбранного класса (конденсаторы, резисторы и т.п.) |

В таблице 31 представлены методы и атрибуты класса Project.

Таблица 31. Методы и атрибуты класса Project

|  |  |
| --- | --- |
| **Наименование** | **Описание** |
| string Path | Путь файла проекта |
| string Name | Наименование проекта |
| Elements ProjectElemets | Список элементов проекта |
| int Time | Время планируемой работы системы |
| int Gamma | Гамма-процент |
| Project(Elements elements) | Конструктор класса |
| SaveProject() | Сохраняет проект |
| SaveProjectAs() | Открывает обозреватель файловой системы для сохранения проекта |
| Project OpenProject() | Открывает обозреватель файловой системы для открытия проекта |

В таблице 32 представлены методы и атрибуты класса Element.

Таблица 32. Методы и атрибуты класса Element

|  |  |
| --- | --- |
| **Наименование** | **Описание** |
| string Name | Наименование элемента |
| double Lyambda | Интенсивность отказов элемента |
| double P | Вероятность безотказной работы элемента |
| char LogicArg | Логическая переменная элемента |
| Element(string name,char logicArg) | Конструктор класса |
| SetLyambdaAndCalculateP(  double lyambda) | Присваивает значение интенсивности отказов элементу и рассчитывает его вероятность безотказной работы |

В таблице 33 представлены методы и атрибуты класса Elements.

Таблица 33. Методы и атрибуты класса Elements

|  |  |
| --- | --- |
| **Наименование** | **Описание** |
| List<Element> ElementsList | Список элементов |
| Elements() | Конструктор класса по умолчанию |
| Elements(List<Element> ElList) | Конструктор класса, принимающий список элементов в качестве аргумента |
| Elements(Elements elements) | Конструктор класса, принимающий собственный объект в качестве аргумента |

В таблице 34 представлены методы и атрибуты класса LogicalMethod.

Таблица 34. Методы и атрибуты класса LogicalMethod

|  |  |
| --- | --- |
| **Наименование** | **Описание** |
| double CalculateLyambda(double P,  int time) | Рассчитывает интенсивность отказов системы |
| double CalculateTsr(double lymbda) | Рассчитывает среднее время наработки до отказа системы |
| double CalculateTy(double T,  int gamma) | Рассчитывает гамма-процентную наработку до отказа системы |
| List<string> DivideFormula(  string formula) | Делит строку с расчетной формулой на несколько частей |
| double CalculateAll(List<string> formulas) | Преобразовывает все части расчетной формулы в алгебраическое выражение. Рассчитывает вероятность безотказной работы системы |
| ReliabilityCalculate(Elements ElementsList, string algebraicForm) | Рассчитывает все показатели надежности технической системы |

В таблице 35 представлены методы и атрибуты класса SDNF.

Таблица 35. Методы и атрибуты класса SDNF

|  |  |
| --- | --- |
| **Наименование** | **Описание** |
| bool Evaluate(String input) | Вычисляет значение логической функции для текущего сочетания значений её аргументов |
| List<String> Digits\_form(List<char> operands, int pos\_index) | Перебирает все возможные сочетания нулей и единиц для каждого из аргументов функции. |
| String GetSDNF(String expr\_str) | Преобразует логическую функцию в СДНФ |

Таблица 35. Продолжение

|  |  |
| --- | --- |
| **Наименование** | **Описание** |
| String Convert\_to\_expr(String expr\_str, out List<char> operands, bool convert\_neg = false) | Преобразует строку с логическими операциями в алгебраическую форму |
| String Convert\_to\_expr(String expr\_str) | Преобразует СДНФ в алгебраическую форму |

В таблице 36 представлены методы и атрибуты класса ReliabilityResults.

Таблица 36. Методы и атрибуты класса ReliabilityResults

|  |  |
| --- | --- |
| **Наименование** | **Описание** |
| double Lymbda | Интенсивность отказов системы |
| double Ty | Гамма-процентная наработка до отказа системы |
| double Tsr | Средняя наработка до отказа системы |
| double P | Вероятность безотказной работы системы |
| ReliabilityResults(double Ty,  double Tsr, double P, double lymbda) | Конструктор класса |

В таблице 37 представлены методы и атрибуты класса ReliabilityResultsContainer.

Таблица 37. Методы и атрибуты класса ReliabilityResultsContainer

|  |  |
| --- | --- |
| **Наименование** | **Описание** |
| ReliabilityResults Results | Результаты расчета надежности технической системы |

В таблице 38 представлены методы и атрибуты класса ExcelReport.

Таблица 38. Методы и атрибуты класса ExcelReport

|  |  |
| --- | --- |
| **Наименование** | **Описание** |
| string PATH | Путь сохранения отчета |
| ReliabilityResults results | Результаты расчета надежности |
| ExcelReport(ReliabilityResults results) | Конструктор класса |
| SaveReport() | Формирует и сохраняет отчет с рассчитанными показателями надежности технической системы в файл формата «Таблица Excel \*.xls» |

В таблице 39 представлены методы и атрибуты класса BD.

Таблица 39. Методы и атрибуты класса BD

|  |  |
| --- | --- |
| **Наименование** | **Описание** |
| BD() | Конструктор класса |
| List<ERI> GetElementsList() | Возвращает список всех элементов из БД |
| List<Class> GetClassList() | Возвращает список всех классов ЭРИ из БД |
| List<ERI> GetListOfCondensators() | Возвращает список всех конденсаторов из БД |
| List<ERI> GetListOfResistors() | Возвращает список всех резисторов из БД |
| List<ERI> GetListOfIMS() | Возвращает список всех интегральных микросхем из БД |
| List<ERI> GetListOfSemiconductor() | Возвращает список всех полупроводниковых приборов из БД |

Таблица 39. Продолжение

|  |  |
| --- | --- |
| **Наименование** | **Описание** |
| List<ERI> GetListOfTransformators() | Возвращает список всех трансформаторов из БД |
| List<ERI> GetListOfOptoelectr() | Возвращает список всех оптоэлектронных приборов из БД |

В таблице 40 представлены методы и атрибуты класса Class.

Таблица 40. Методы и атрибуты класса Class

|  |  |
| --- | --- |
| **Наименование** | **Описание** |
| int ID | Идентификационный номер Класса ЭРИ |
| string Name | Наименование класса ЭРИ |
| ICollection<ERI> ERI | Коллекция всех ЭРИ, относящихся к данному классу |
| Class() | Конструктор класса |

В таблице 41 представлены методы и атрибуты класса ERI.

Таблица 41. Методы и атрибуты класса ERI

|  |  |
| --- | --- |
| **Наименование** | **Описание** |
| int ID | Идентификационный номер ЭРИ |
| string Name | Наименование ЭРИ |
| double Lyambda | Интенсивность отказов ЭРИ |
| int IDClass | Внешний ключ к сущности Class |
| Class Class | Класс, к которому относится данное ЭРИ |

**4.2. Разработка пользовательского интерфейса программы для автоматизации прогнозирования надежности сложных технических систем**

На рисунке 21 показан интерфейс главной формы.

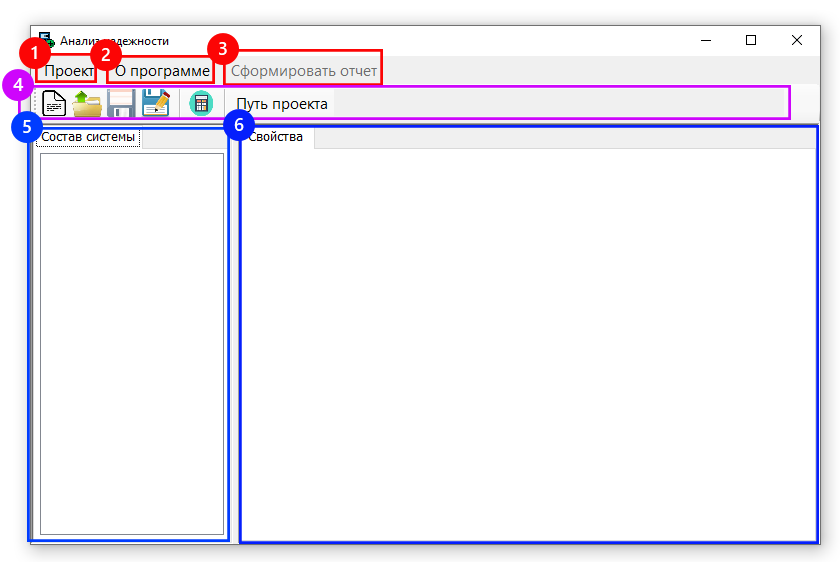


Рис. 21. Интерфейс главной формы

Описание элементов интерфейса главной формы:

1. «Проект». При нажатии на этот пункт меню откроется подменю со следующими опциями: «Создать новый», «Открыть», «Сохранить» и «Сохранить как» (см. рис. 22).

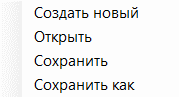


Рис. 22. Подменю пункта «Проект»

При нажатии на опцию «Создать новый» появится диалоговое меню (см. рис. 23), в котором пользователю будет предложено ввести наименование проекта. При нажатии на кнопку «ОК» добавится новый проект с введенным названием. Нажатие на кнопку «Отмена» приведет к закрытию диалогового окна, проект не будет создан.

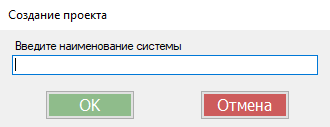


Рис. 23. Создание нового проекта

При нажатии на опцию «Открыть» откроется диалоговое меню (см. рис. 24), в котором пользователь может выбрать существующий файл с проектом.

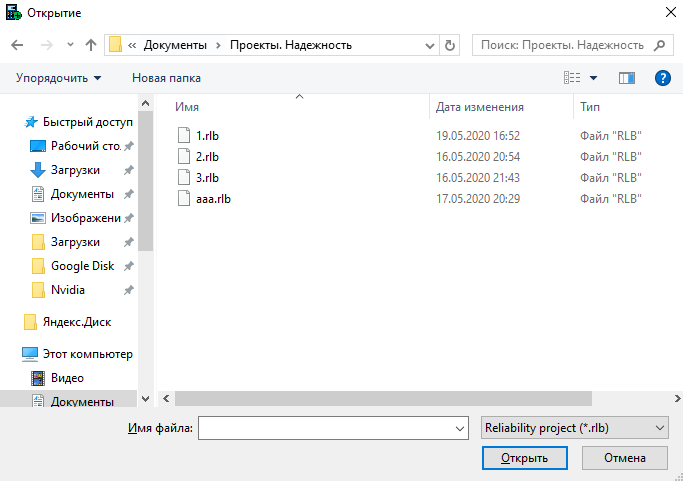


Рис. 24. Открытие проекта

При нажатии на опцию «Сохранить» программа перезапишет файл с текущим проектом. Если же текущий проект никогда не сохранялся, откроется диалоговое окно (см. рис. 25), в котором пользователь может выбрать путь сохранения проекта.

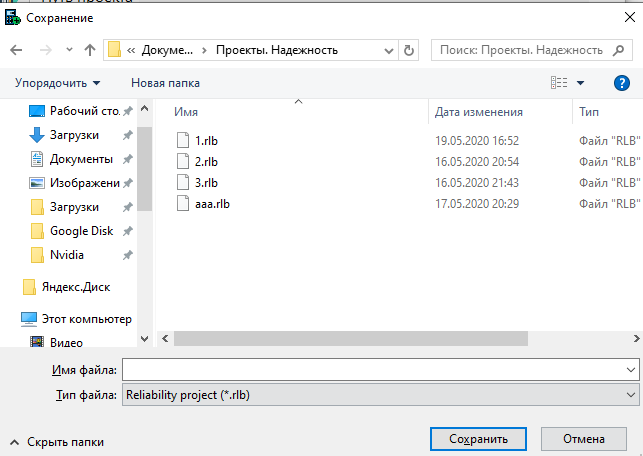


Рис. 25. Сохранение проекта

При нажатии на опцию «Сохранить как» программа откроет диалоговое окно (см. рис. 25), в котором пользователю будет предложено выбрать путь сохранения проекта.

2. «О программе». При нажатии на этот пункт меню откроется окно с информацией о программе (см. рис. 26).

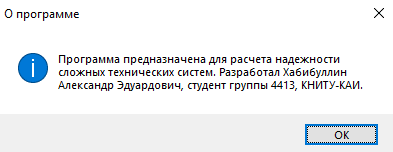


Рис. 26. Описание программы

3. «Сформировать отчет». При нажатии на этот пункт меню сформируется отчет с рассчитанными показателями надежности технической системы. Данная опция будет заблокирована до тех пор, пока пользователь не произведет расчет надежности.

4. Панель инструментов. Первые четыре кнопки (см. рис. 27) по функционалу аналогичны опциям из пункта 1 «Создать новый», «Открыть», «Сохранить» и «Сохранить как» соответственно.



Рис. 27. Управление проектом

Строка «Путь проекта» будет показывать путь до файла с текущим (открытым) проектом.

Кнопка «Расчет надежности» (см. рис. 28) инициализирует процесс расчета надежности технической системы.

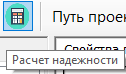


Рис. 28. Кнопка «Расчет надежности»

5. Дерево проекта. В качестве основного узла дерева выступает сам проект (техническая система). Наименование основного узла соответствует наименованию проекта. В качестве дочерних узлов выступают элементы технической системы.

6. Панель с тремя вкладками (две из которых будут доступны только после расчета надежности). При запуске программы доступна лишь одна вкладка – «Свойства», в которой пользователь может редактировать свойства проекта или элемента системы, в зависимости от выделенного узла в дереве проекта.

Для начала работы необходимо открыть существующий проект или создать новый.

Создадим новый проект. Для этого в главном окне программы нажмем на «Проект-Создать новый». Откроется диалоговое окно, в котором необходимо ввести наименование проекта (см. рис. 29).

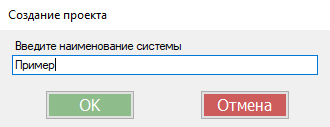


Рис. 29. Создание нового проекта

После ввода наименования проекта, нажимаем на кнопку «ОК». В дереве проекта появится созданный проект (см. рис. 30).

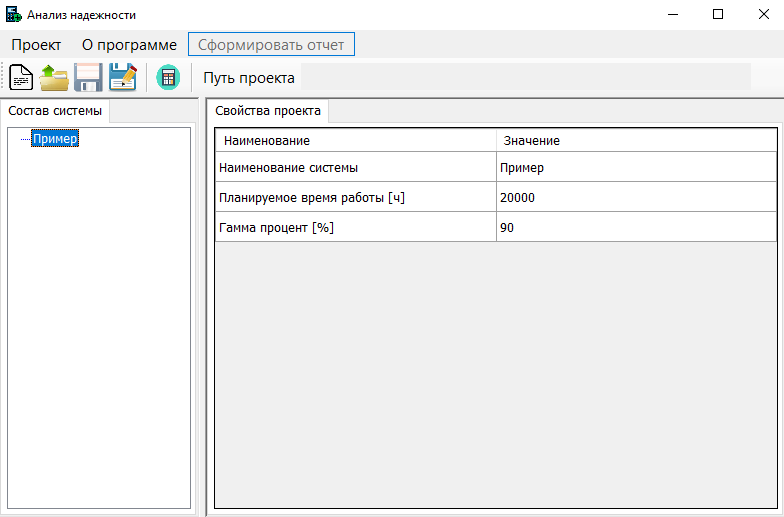


Рис. 30. Главное окно после создания проекта

Во вкладке «Свойства проекта» (см. рис. 31) можно отредактировать следующие параметры проекта: наименование системы (проекта), планируемое время работы и гамма-процент.

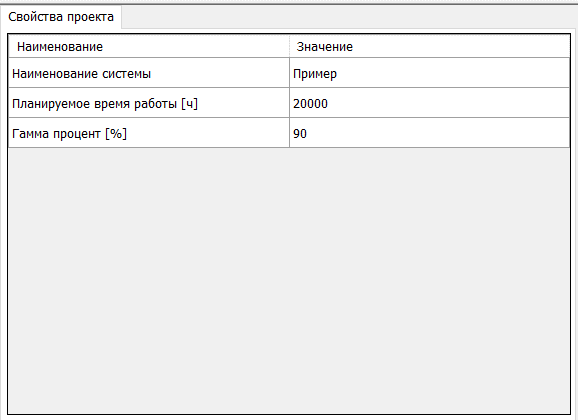


Рис. 31. Редактирование параметров проекта

Добавим новый элемент на схему. Для этого нажмем по названию проекта правой кнопкой мыши и выберем пункт «Добавить компонент – Новый элемент» (см. рис. 32).

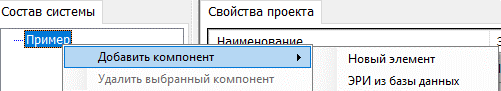


Рис. 32. Добавление нового элемента в проект

В результате, появится диалоговое окно, в котором необходимо ввести наименование элемента (см. рис. 33). После ввода нажимаем кнопку «ОК».

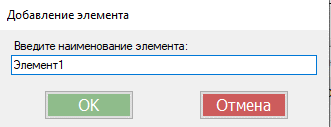


Рис. 33. Ввод наименования нового элемента

Новый элемент появится в дереве проекта (см. рис. 34). Буква в скобках справа от наименования элемента – его логическая переменная.

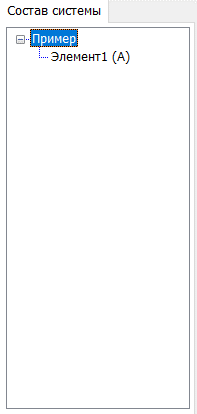


Рис. 34. Дерево проекта после добавления элемента

После нажатия на наименование элемента, появится вкладка с редактированием данных об элементе. Если не ввести значение интенсивности отказов элемента, по умолчанию оно будет равно нулю. Поэтому введем интенсивность отказов для добавленного элемента. Вероятность безотказной работы будет рассчитана автоматически (см. рис. 35).

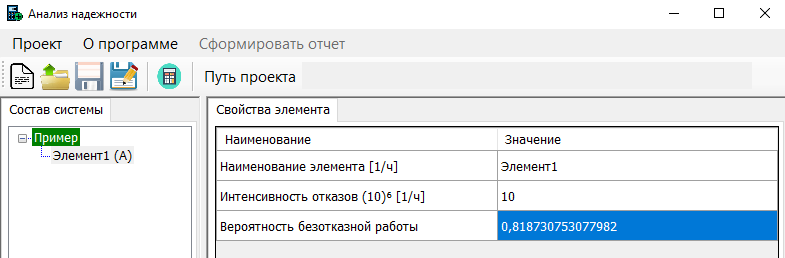


Рис. 35. Редактирование параметров элемента

Добавим в проект ЭРИ из базы данных. Для этого нажмем правой кнопкой мыши по проекту в дереве и выберем пункт «Добавить компонент-ЭРИ из базы данных» (см. рис. 36).

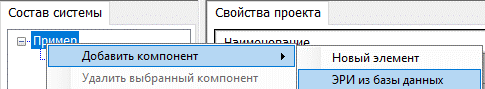


Рис. 36. Добавление ЭРИ из базы данных

Откроется окно для выбора ЭРИ. Здесь необходимо выбрать класс ЭРИ и сам элемент (см. рис. 37). Как только выбор будет осуществлен, необходимо нажать на кнопку «ОК».

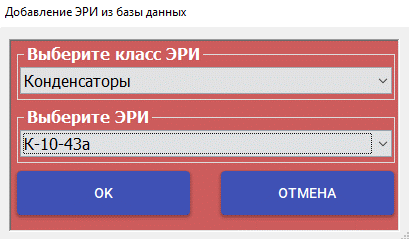


Рис. 37. Выбор ЭРИ

Состав технической системы после добавления ЭРИ представлен на рис. 38. По необходимости пользователь может отредактировать параметры добавленного ЭРИ во вкладке «Свойства элемента».

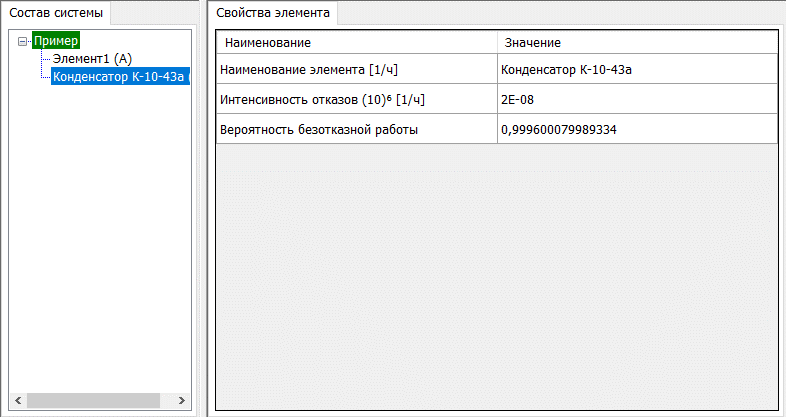


Рис. 38. Состав технической системы после добавления ЭРИ из БД

Сохраним проект, нажав на соответствующую кнопку на панели инструментов (см. рис. 39.)



Рис. 39. Кнопка для сохранения проекта на панели инструментов

Поскольку проект ранее не сохранялся, программа откроет диалоговое окно, в котором необходимо выбрать путь сохранения проекта и ввести наименование файла (см. рис. 40). После выбора директории и ввода наименования файла с проектом, нажимаем на кнопку «Сохранить».

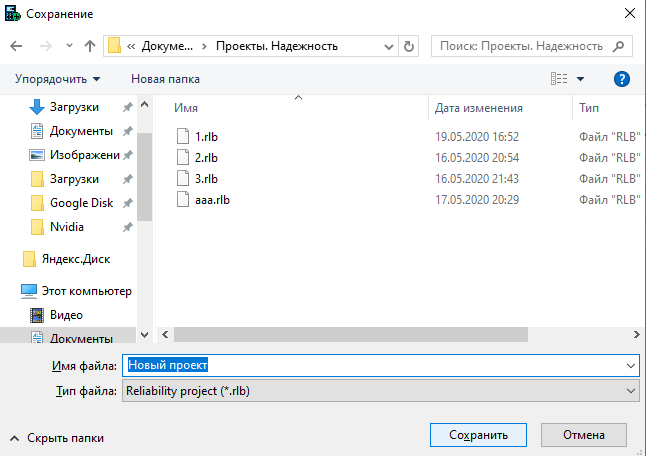


Рис. 40. Сохранение проекта

Проект сохранен. В строке «Путь проекта» на панели инструментов появится полный путь до сохраненного файла с проектом (см. рис. 41).



Рис. 41. Путь до сохраненного файла с проектом

Произведем расчет надежности технической системы. Для этого нажмем на кнопку «Расчет надежности» на панели инструментов (см. рис. 42).



Рис. 42. Кнопка для инициализации процесса расчета надежности

Появится окно для ввода логической функции работоспособности системы (см. рис. 43).



Рис. 43. Окно для ввода логической функции

Зеленые кнопки – доступны для нажатия в данный момент. Поскольку в системе сейчас всего 2 элемента, логическими переменными которых являются «А» и «B» соответственно, для нажатия доступны кнопки именно с этими логическими переменными.

Красные кнопки – недоступны для нажатия до тех пор, пока пользователь не добавит больше элементов в систему.

Серые кнопки:

1. «^» - знак конъюнкции;
2. «v» - знак дизъюнкции;
3. «<-» - перемещает каретку ввода на одну позицию влево;
4. «->» - перемещает каретку ввода на одну позицию вправо;
5. «Back» - удаляет элемент, который находится перед кареткой ввода;
6. «Clear» - очищает текстовое поле.

Введем логическую функцию работоспособности системы и нажмем кнопку «ГОТОВО» (см. рис. 44).



Рис. 44. Окно с введенной логической функцией

Программа осуществила расчет надежности. На главной форме программы появились две вкладки «Параметры безотказности» и «Пошаговое решение» (см. рис. 45).

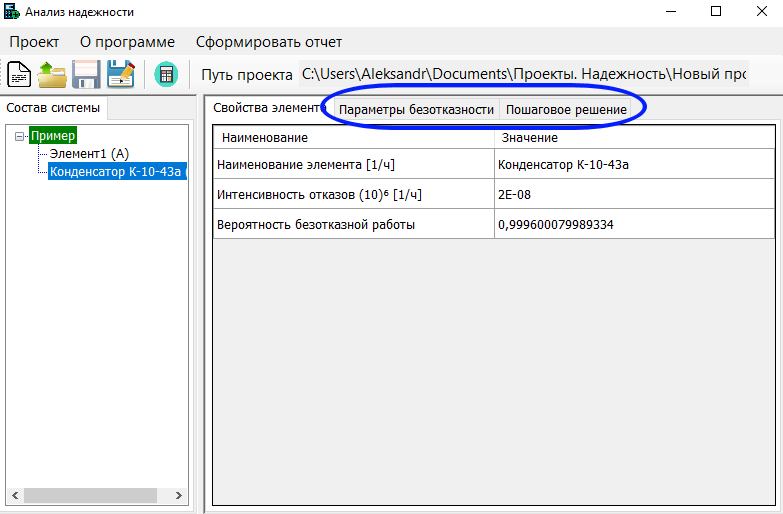


Рис. 45. Главная форма после расчета надежности

На вкладке «Параметры безотказности» можно посмотреть на результаты расчета надежности технической системы (см. рис. 46).

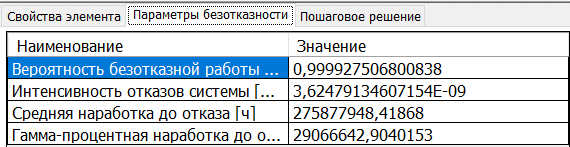


Рис. 46. Результаты расчета

На вкладке «Пошаговое решение» можно ознакомиться с процессом расчета надежности по шагам (см. рис. 47).

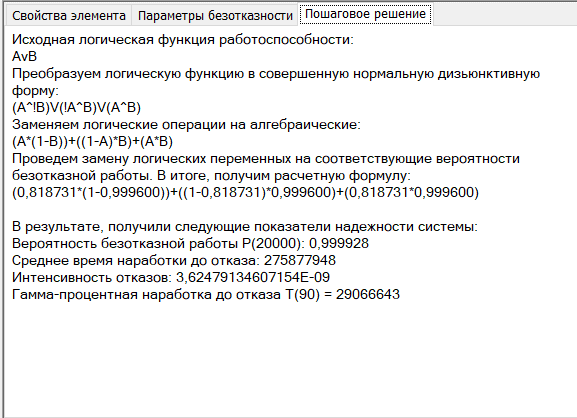


Рис. 47. Пошаговое решение

Теперь мы можем сформировать отчет. Для этого нажмем на пункт меню «Сформировать расчет» (см. рис. 48).

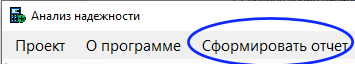


Рис. 48. Меню программы

Программа формирует и открывает отчет с рассчитанными показателями надежности в формате файлов Excel (см. рис. 49).

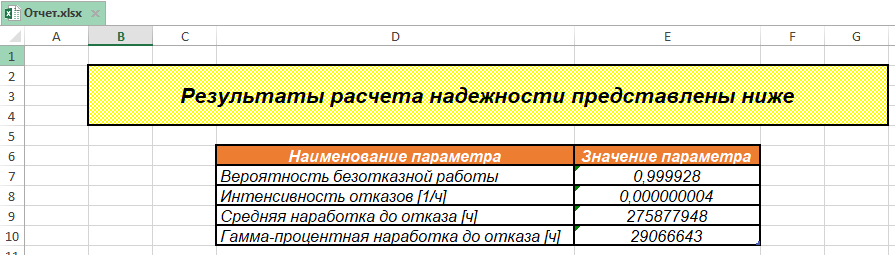


Рис. 49. Сформированный отчет

**4.3. Примеры решения задачи прогнозирования надежности сложных технических систем**

Рассмотрим работоспособность программы на нескольких примерах.

1) Сначала проведем проверку на контрольном примере из раздела 2.3. Исходные данные те же (таблица 3), блок-схема надежности представлена на рис. 11.

Создадим проект и добавим в него четыре элемента, имеющихся на блок-схеме. Введем время планируемой работы t = 20000ч и гамма-процент = 95% (см. рис. 50).

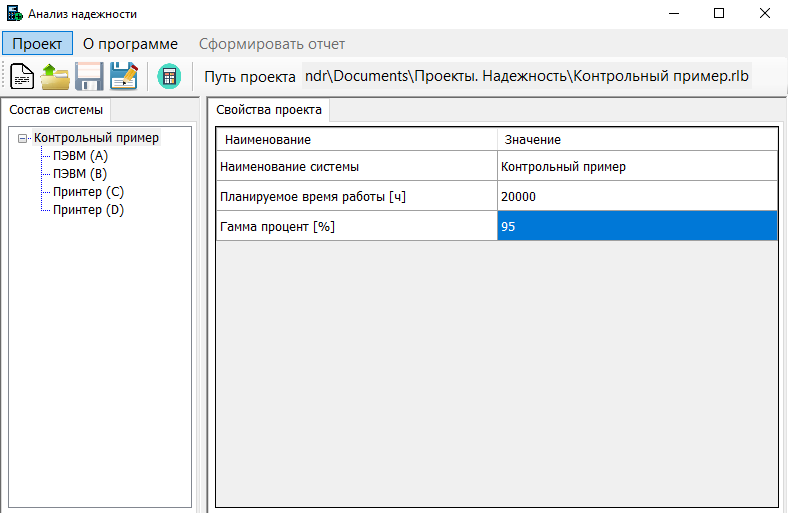


Рис. 50. Параметры проекта для контрольного примера

Каждому элементу, согласно таблице 3, проставим его интенсивность отказов (см. рис. 51).

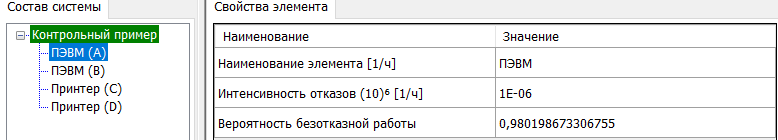


Рис. 51. Ввод интенсивности отказов

Нажмем на кнопку «Расчет надежности» и введем логическую функцию работоспособности системы, после чего нажимаем на кнопку «ГОТОВО» (см. рис. 52).

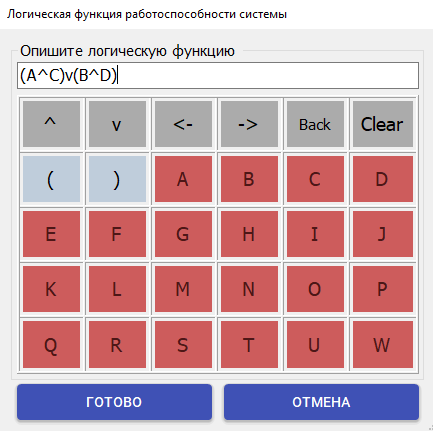


Рис. 52. Ввод логической функции работоспособности системы

Результаты расчета программы представлены в виде пошагового решения на рис. 53.

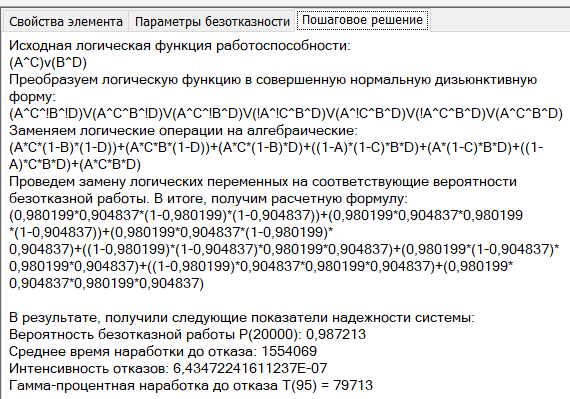


Рис. 53. Результаты расчета

Рассчитанные программой показатели надежности совпадают с результатами ручного расчета из раздела 2.3.

2) Проведем автоматизированный расчет надежности вентилятора на асинхронном электродвигателе, схема которого представлена на рис. 54.

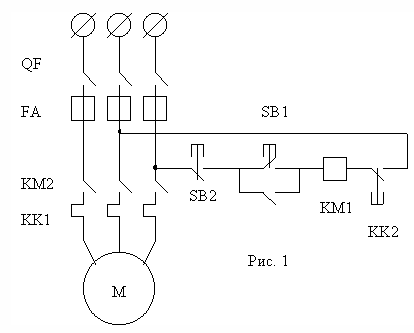


Рис. 54. Вентилятор на асинхронном электродвигателе

Составим блок-схему надежности. Результат представлен на рис. 55.

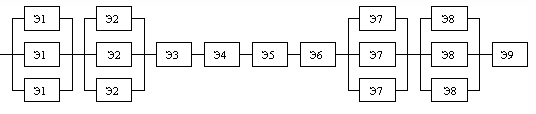


Рис. 55. Блок-схема надежности вентилятора

Исходные данные для расчета представлены в таблице 42. Время планируемой работы t = 60000ч, гамма процент = 95%.

Таблица 42. Исходные данные для расчета

|  |  |  |
| --- | --- | --- |
| **Наименование элемента** | **Обозначение элемента**  **на расчетной схеме** | **Интенсивность отказов элемента, 10-6**  **[1/ч]** |
| QF | Э1 | 3 |
| FA | Э2 | 15 |
| KK2 | Э3 | 0,1 |
| KM1 | Э4 | 7,5 |
| SB1 | Э5 | 0,05 |
| SB2 | Э6 | 0,6 |
| KM2 | Э7 | 0,002 |
| KK1 | Э8 | 0,8 |
| M | Э9 | 0,0001 |

Создадим новый проект, добавим в него все элементы и введем для каждого элемента интенсивность отказов. Результат проделанных операций представлен на рис. 56.

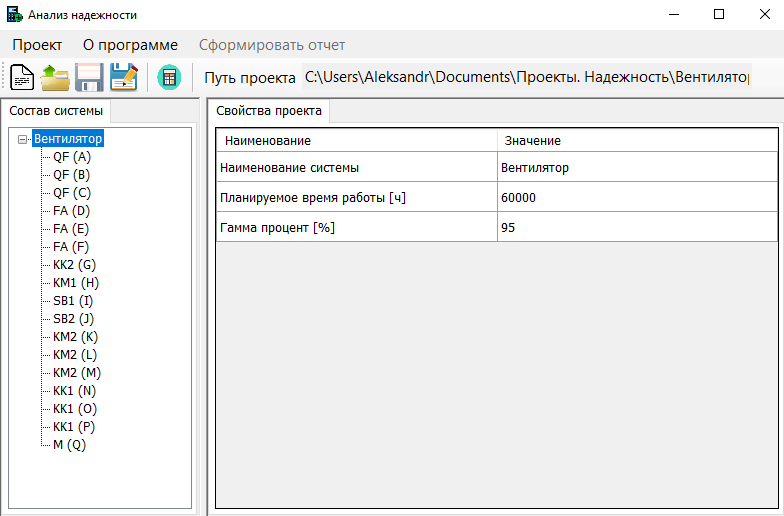


Рис. 56. Главная форма программы после добавления всех элементов

Нажмем на кнопку «Расчет надежности» и введем логическую функцию работоспособности системы (см. рис. 57).



Рис. 57. Логическая функция работоспособности вентилятора

После ввода логической функции нажимаем на кнопку «ГОТОВО». Программа осуществила расчет надежности. Результаты расчета представлены на рис. 58.

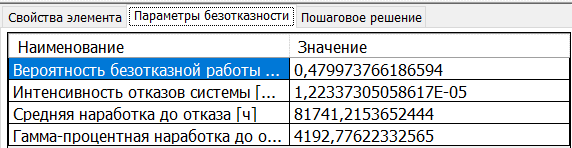


Рис. 58. Результаты расчета

Сформируем отчет, нажав на опцию меню «Сформировать отчет». Сформированный отчет представлен на рис. 59.

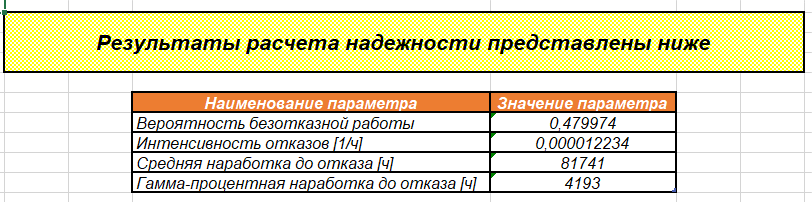


Рис. 59. Отчет о надежности вентилятора

**Выводы к главе 4**

1. Сформулированы основные функциональные требования к программе;

2. разработано программное обеспечение, позволяющее проводить расчет надежности сложных технических систем с использованием логико-вероятностного метода;

3. написано руководство пользователя для программы;

4. доказана работоспособность программы на двух примерах.

**ЗАКЛЮЧЕНИЕ**

В ходе выполнения выпускной квалификационной работы были достигнуты следующие результаты:

1. проведен анализ предметной области прогнозирования надежности сложных технических систем;
2. сформулированы содержательная и математическая постановки задачи прогнозирования надежности сложных технических систем;
3. разработан логико-вероятностный алгоритм прогнозирования надежности сложных технических систем;
4. разработано информационное обеспечение автоматизированного прогнозирования надежности сложных технических систем;
5. разработано программное обеспечение автоматизированного прогнозирования надежности сложных технических систем в среде Visual Studio 2019 с использованием языка C#;
6. доказана работоспособность программы на двух примерах.

Применение логико-вероятностного метода для расчета надежности сложных технических систем без применения ЭВМ – нецелесообразно, поскольку преобразования логической функции работоспособности являются очень трудоемким процессом. Особенно это заметно при преобразовании логической функции в СДНФ при большом количестве параллельных структур.

Если техническая система состоит из небольшого числа элементов, и автоматизация расчета надежности не планируется, следует вместо совершенной нормальной дизъюнктивной формы использовать другую форму перехода к замещению (например, бесповторную форму в базисе конъюнкция-отрицание), так как форма представления СДНФ является очень громоздкой.

Несмотря на все недостатки, логико-вероятностный метод является эффективным методом анализа надежности сложных технических систем, поскольку он позволяет анализировать надежность любой структурно-сложной технической системы.

**СПИСОК ЛИТЕРАТУРЫ**

|  |  |
| --- | --- |
| 1. | Антимиров В.М. Поколения УВС // Вопросы атомной науки и техники. Научно-технический сборник. — М.: ФГУП НИИП, 2012. — Вып. 2. — С. 38−48. |
| 2. | Базовский И. Надѐжность. Теория и практика / Базовский И., (пер. с англ.) - М.: Мир, 1965. -375 с. |
| 3. | Вентцель Е.С., Теория случайных процессов и ее инженерные приложения: учебное пособие / Е. С. Вентцель, Л. А. Овчаров. — 5-е изд., стер. — М. : КНОРУС, 2013. — 448 с. |
| 4. | Викторова В.С., Проектный анализ контроля пригодности технических систем (теория, методы расчета, программное обеспечение) / Викторова В.С., Степанянц А.С. – Научное издание. – М.:ИПУ РАН, 2010. – 71с. |
| 5. | Городецкий, А. Е. Имитационное моделирование развития аварийных  ситуаций в энергетических установках. / А. Е. Городецкий, В. Г. Курбанов, И. Л. – 141 с. |
| 6. | Гнеденко Б. В., Математические методы в теории надежности / Гнеденко Б. В., Беляев Ю. К., Соловьев А. Д. – М.: Наука, 1965. – 524 с. |
| 7. | Голинкевич, Т. А. Прикладная теория надежности : учеб. для ву-  зов / Т. А. Голинкевич. – М. : Высш. шк., 1985. – 168 с. |
| 8. | ГОСТ 27.002-2015. Надежность в технике: термины и определения/ Межгос. совет по стандартизации, метрологии и сертификации. — М.: Изд-во стандартов, 2015. – 28 с. |
| 9. | ГОСТ Р 51901.14-2007. Менеджмент риска: структурная схема надежности и булевы методы / Межгос. совет по стандартизации, метрологии и сертификации. — М.: Изд-во стандартов, 2007. – 28 с. |
| 10. | Дроздова А. А., Практикум по дисциплине «надежность информационных систем» / Дроздова А. А., Нечипорук Д. В., Ростов-на-Дону: ДГТУ, 2016 – 71 с. |
| 11. | Дубаренко, В. В. Об одном методе вычисления вероятностей логических функций. / В. В. Дубаренко, В. Г. Курбанов, А. Ю. Кучмин // Информационно-управляющие системы. — 2010. — №5. — C. 2–7. |
| 12. | Жаднов В.В., Имитационное моделирование в задачах оценки надежности отказоустойчивых электронных средств / Жаднов В.В., Тихменев А.Н. |
| 13. | Жаднов В.В., Полесский С.Н., Тихменев А.Н. Разработка моделей надежности для проектных исследований надежности радиоэлектронной аппаратуры. / Радиовысотометрия-2010: Сб. тр. Третьей Всероссийской научно-технической конференции. // Под ред. А.А. Иофина, Л.И. Пономарева. – Екатеринбург: Из-во «Форт Диалог-Исеть», 2010. – с. 200-201. |
| 14. | Зиняков, В. Ю. Управление живучестью системы с использованием  логико-вероятностного прогнозирования. / В. Ю. Зиняков, В. Г. Курбанов // Завалишинские чтения'15 сборник докладов. / Санкт-Петербургский государственный университет аэрокосмического приборостроения - СПб, 2015. — C. 146–152. |
| 15. | Майн Х., Марковские процессы принятия решений / Майн Х., Осаки С. М.: Наука, 1977. |
| 16. | Надежность технических систем: Справочник / Ю. К. Беляев, В. А. Богатырев, В. В. Болотин и др.; Под. ред. И. А. Ушакова. – М.: Радио и связь, 1985. – 608 с. |
| 17. | Нечипоренко, В. И. Структурный анализ систем. / В. И. Нечипоренко;  Главпочтамт. — М. : Издательство «Советское радио», 1977. |
| 18. | Обеспечение надёжности электронных средств : конспект лек-  ций / Владим. гос. ун-т. им. А. Г. и Н. Г. Столетовых ; сост. : Е. Н. Та-  лицкий, С. В. Шумарин. – Владимир : Изд-во ВлГУ, 2013. – 40 с. |
| 19. | Основы надежности электронных средств: учеб. пособие для студентов  всех форм обучения / А.В. Баранова, Н.П. Ямпурин; НГТУ (АПИ) –  Нижний Новгород: НГТУ, 2005. – 97 с. |
| 20. | Острейковский В. А. Теория надежности: Уч. для вузов. – М. Высш. шк., 2003. – 463 с. |
| 21. | Половко А. М. Основы теории надежности. – Спб.: БХВ-Петербург, 2006. – 560 с. |
| 22. | Расчёт показателей надёжности радиоэлектронных средств : учеб.-метод. пособие / С. М. Боровиков, И. Н. Цырельчук, Ф. Д. Троян ; под ред. С. М. Боровикова. – Минск : БГУИР, 2010. – 68 с. : ил. |
| 23. | Рябинин, И. А. Надежность и безопасность структурно-сложных  систем. / И. А. Рябинин. — СПб : Политехника, 2000. — 248с. |
| 24. | Строганов А. В., Жданов В. К., Полесский С. М. Обзор программных комплексов по расчету надежности сложных технических систем / под ред. Д. Д. Краснова. М.: ВШЭ, 2007. 185 с. |
| 25. | Ховард Р. А. Динамическое программирование и марковские процессы / Ховард Р. А., М.: Сов. радио, 1964. |
| 26. | Шаламов А. В., Обзор программных комплексов расчета надежности технических систем / Шаламов А. В., Соловьев В. А., М.: Молодежный научно-технический вестник, 2016 – 9 с. |
| 27. | Шубинский, И. Б. Функциональная надежность информационных систем. Методы анализа / И. Б. Шубинский. – М.: «Журнал Надежность», 2012, – 296 с., ил. |
| 28. | Шубинский, И. Б. Структурная надежность информационных систем. Методы анализа / И. Б. Шубинский. – М.: «Журнал Надежность», 2012, – 216 с., ил. |
| 29. | Юрков, Н. К. Технология производства электронных средств : учебник / Н. К. Юрков. — 2-е изд., испр., доп. — Санкт-Петербург : Лань, 2014. — 480 с. — ISBN 978-5-8114-1552-6. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/41019 (дата обращения: 28.05.2020). — Режим доступа: для авториз. пользователей. |

**ПРИЛОЖЕНИЕ А. Глоссарий**

Электронное средство представляет собой сложную техническую систему, в основе функционирования которой лежат процессы передачи, извлечения и обработки информации, связанные с преобразованием и передачей электромагнитной энергии.

Логико-вероятностный метод – метод анализа надежности, при котором структура системы описывается средствами математического аппарата бинарной алгебры логики, а количественная оценка надежности производится с помощью теории вероятностей.

Имитационное моделирование (метод Монте-Карло) - численный метод проведения на ЭВМ вычислительных экспериментов с математическими моделями, имитирующими поведение реальных объектов и процессов.

Техническое задание (ТЗ) − документ, содержащий требования заказчика к объекту закупки, определяющие условия и порядок ее проведения для обеспечения государственных или муниципальных нужд, в соответствии с которым осуществляются поставка товара, выполнение работ, оказание услуг и их приемка.

Отказ – полная или частичная потеря работоспособности изделием вследствие ухода одного или более функциональных параметров за пределы установленных норм.

Под безотказностью понимают свойство изделия непрерывно сохранять работоспособное состояние в течение заданного времени.

Расчет надежности - это процедура определения значений показателей надежности объекта с использованием методов, основанных на их вычислении по справочным данным о надежности элементов объекта, по данным о надежности объектов-аналогов, данным о свойствах материалов и другой информации, имеющейся к моменту расчета.

Вероятность безотказной работы - вероятность того, что за указанное время работы отказ устройства не произойдет.

Вероятность отказа - вероятность того, что за указанное время работы произойдет отказ устройства.

Средняя наработка до отказа – среднее время, в течение которого устройство будет исправно работать (до возникновения первого отказа);

Гамма-процентная наработка до отказа - наработка, в течение которой отказ устройства не возникнет с заданной вероятностью.

Интенсивность отказов – условная плотность вероятности возникновения отказа объекта, определяемая при условии, что до рассматриваемого момента времени отказ не возник. Численно равна числу отказов в единицу времени, отнесенное к числу узлов, безотказно проработавших до этого времени.

**ПРИЛОЖЕНИЕ Б. Листинг программы**

using org.mariuszgromada.math.mxparser;

using System;

using System.Collections.Generic;

using System.Linq;

namespace Reliability

{

/// <summary>

/// Логико-вероятностный метод

/// </summary>

class LogicalMethod

{

public LogicalMethod()

{

}

/// <summary>

/// Рассчитывает интенсивность отказов системы

/// </summary>

/// <param name="P"> Вероятность безотказной работы системы</param>

/// <param name="time"> Время работы системы </param>

/// <returns></returns>

private double CalculateLyambda(double P,int time)

{

return -(Math.Log(P)/(double)time);

}

/// <summary>

/// Рассчитывает среднее время наработки системы до отказа

/// </summary>

/// <param name="lymbda"> Интенсивность отказов</param>

/// <returns></returns>

private double CalculateTsr(double lymbda)

{

return 1.0 / lymbda;

}

/// <summary>

/// Рассчитывает Гамма-процентную наработку до отказа всей системы

/// </summary>

/// <param name="T"> Время работы системы</param>

/// <param name="gamma"> Гамма-процент</param>

/// <returns></returns>

private double CalculateTy(double T, int gamma)

{

double osnovanie = ((double)gamma / 100.0);

return -T \* Math.Log(osnovanie);

}

/// <summary>

/// Делит расчетную формулу на несколько частей по знаку +

/// </summary>

/// <param name="formula"> Итоговая формула для расчета</param>

/// <returns></returns>

private List<string> DivideFormula(string formula)

{

int startIndex;

int endIndex;

List<string> results = new List<string>();

List<int> indices = new List<int>() { 0 };

char search = '+';

// Ищем все вхождения знака ПЛЮС

int index = formula.IndexOf(search);

if(index < 0)

{

startIndex = 0;

endIndex = formula.Length;

results.Add(formula.Substring(startIndex, endIndex));

return results;

}

while (index > -1)

{

indices.Add(index);

index = formula.IndexOf(search, index + 1);

}

int count = indices.Count();

// Копируем каждую последовательность до очередного знака + в список

for (int i = 0; i < count - 1; i++)

{

startIndex = indices[i] != 0 ? indices[i] + 1 : 0;

endIndex = indices[i + 1];

results.Add(formula.Substring(startIndex, endIndex - startIndex));

}

results.Add(formula.Substring(indices[count - 1] + 1, formula.Length - indices[count - 1] - 1));

return results;

}

/// <summary>

/// Осуществляет последовательное сложение всех формул и рассчитывает вероятность безотказной работы системы

/// </summary>

/// <param name="formulas"> Список всех формул для их поочередного сложения</param>

/// <returns></returns>

private double CalculateAll(List<string> formulas)

{

List<double> listofValues = new List<double>(formulas.Count);

foreach (string str in formulas)

{

Expression parser = new Expression(str);

listofValues.Add(parser.calculate());

}

return listofValues.Sum();

}

/// <summary>

/// Рассчитывает вероятность безотказной работы системы

/// </summary>

/// <param name="ElementsList">Список элементов системы</param>

/// <param name="algebraicForm">Алгебраическая форма СДНФ</param>

/// <returns></returns>

public void ReliabilityCalculate(Elements ElementsList, string algebraicForm)

{

string finalStr = algebraicForm;

double Tsr;

double Ty;

double lymbda;

double P;

string formula = algebraicForm;

foreach(Element element in ElementsList.ElementsList)

{

finalStr = finalStr.Replace(element.LogicArg.ToString(), $"{element.P:0.000000}");

formula = formula.Replace(element.LogicArg.ToString(),$"{element.P}");

}

DataContainer.FinalAlgebraicFormula = finalStr;

formula = formula.Replace(",", ".");

List<string> expressions = DivideFormula(formula);

P = CalculateAll(expressions);

lymbda = CalculateLyambda(P,DataContainer.Time);

Tsr = CalculateTsr(lymbda);

Ty = CalculateTy(Tsr,DataContainer.GammaProject);

ReliabilityResultsContainer.Results = new ReliabilityResults(Ty,Tsr,P,lymbda);

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Reliability

{

static class SDNF

{

/// <summary>

/// Вычисляет значение логической функции для текущего сочетания значений её аргументов.

/// В случае, если оно истинно, то на основе значений всех аргументов формируется очередная по счёту дизъюнкция для записи СДНФ

/// </summary>

private static bool Evaluate(String input)

{

String expr = "(" + input + ")";

Stack<char> ops = new Stack<char>();

Stack<bool> vals = new Stack<bool>();

for (int i = 0; i < expr.Length; i++)

{

char ch = expr[i];

if (ch.Equals('(')) ops.Push(ch);

else if (ch.Equals('+')) ops.Push(ch);

else if (ch.Equals('\*')) ops.Push(ch);

else if (ch.Equals('!')) ops.Push(ch);

else if (ch.Equals(')'))

{

int count = ops.Count;

if (count == 0)

continue;

char op = ops.Pop();

while (count > 0)

{

if (op.Equals('('))

break;

bool v = vals.Pop();

if (op.Equals('+'))

v = vals.Pop() || v;

else if (op.Equals('!'))

v = !v;

bool inner\_oper = false;

do

{

count--;

if (op.Equals('\*'))

{

v = vals.Pop() && v;

if (count > 0)

op = ops.Pop();

inner\_oper = true;

}

} while (count > 0 && op.Equals('\*'));

vals.Push(v);

if (count > 0 && !inner\_oper)

op = ops.Pop();

}

}

else vals.Push(ch == '1');

}

return vals.Pop();

}

/// <summary>

/// Перебирает все возможные сочетания нулей и единиц для каждого из аргументов функции.

/// Функция возвращает список всех возможных значений аргумента на данном шаге,

/// каждое из которых затем добавляется к значениям аргументов, полученных на предыдущем шаге

/// </summary>

/// <param name="operands">Список аргументов</param>

/// <param name="pos\_index">Позиция нужного аргумента</param>

private static List<String> Digits\_form(List<char> operands, int pos\_index)

{

if (pos\_index < operands.Count)

{

String operand = operands[pos\_index].ToString();

List<String> output\_forms = Digits\_form(operands, pos\_index + 1);

List<String> result\_forms = new List<string>();

foreach (String output\_form in output\_forms)

for (byte value = 0; value <= 1; value++)

result\_forms.Add(output\_form.Replace(operand, value.ToString()));

return result\_forms;

}

else

return new List<string>() { String.Concat(operands.Select(s => s.ToString())) };

}

/// <summary>

/// Преобразует логическую функцию в СДНФ

/// </summary>

public static String GetSDNF(String expr\_str)

{

List<char> operands;

String converted\_expr = Convert\_to\_expr(expr\_str, out operands);

StringBuilder result = new StringBuilder();

List<String> output\_forms = Digits\_form(operands, 0);

int count = 0;

bool value;

foreach (String output\_form in output\_forms)

{

String input = converted\_expr;

for (int index = 0; index < operands.Count; index++)

input = input.Replace(operands[index], output\_form[index]);

value = Evaluate(input);

if (value)

{

if (count > 0)

result.Append('V');

result.Append('(');

for (int index = 0; index < operands.Count; index++)

{

if (output\_form[index] == '0')

result.Append('!');

result.Append(operands[index]);

if (index < operands.Count - 1)

result.Append('^');

}

result.Append(')');

count++;

}

}

return result.ToString();

}

/// <summary>

/// Преобразует СДНФ в расчетную формулу

/// </summary>

private static String Convert\_to\_expr(String expr\_str, out List<char> operands, bool convert\_neg = false)

{

operands = new List<char>();

StringBuilder result = new StringBuilder();

bool scan\_neg = false;

for (int index = 0; index < expr\_str.Length; index++)

{

char ch = expr\_str[index];

switch (ch)

{

case '(':

case ')':

result.Append(ch);

break;

case ' ':

break;

case 'v':

case 'V':

result.Append('+');

break;

case '^':

result.Append('\*');

break;

case '!':

case '¬':

case '┐':

if (convert\_neg)

{

result.Append("(1-");

scan\_neg = true;

}

else

result.Append("!");

break;

default:

if (!operands.Contains(ch))

operands.Add(ch);

result.Append(ch);

if (scan\_neg)

{

result.Append(')');

scan\_neg = false;

}

break;

}

}

return result.ToString();

}

/// <summary>

/// Преобразует СДНФ в алгебраическую форму

/// </summary>

public static String Convert\_to\_expr(String expr\_str)

{

List<char> operands;

return Convert\_to\_expr(expr\_str, out operands, true);

}

}

}

using System.Collections.Generic;

using System.Linq;

namespace Reliability.Database

{

/// <summary>

/// Класс для работы с базой данных

/// </summary>

class BD

{

public ReliabilityContext context { get; }

public BD()

{

context = new ReliabilityContext();

}

public List<ERI> GetElementsList()

{

var elements = context.ERI.ToList();

return elements;

}

public List<Class> GetClassList()

{

var classes = context.Class.ToList();

return classes;

}

public List<ERI> GetListOfCondensators()

{

IQueryable<ERI> query = context.ERI.Where(c => c.IDClass == 1);

List<ERI> condensators = query.ToList();

return condensators;

}

public List<ERI> GetListOfResistors()

{

IQueryable<ERI> query = context.ERI.Where(c => c.IDClass == 2);

List<ERI> resistors = query.ToList();

return resistors;

}

public List<ERI> GetListOfIMS()

{

IQueryable<ERI> query = context.ERI.Where(c => c.IDClass == 3);

List<ERI> IMS = query.ToList();

return IMS;

}

public List<ERI> GetListOfSemiconductor()

{

IQueryable<ERI> query = context.ERI.Where(c => c.IDClass == 4);

List<ERI> semiconductor = query.ToList();

return semiconductor;

}

public List<ERI> GetListOfTransformators()

{

IQueryable<ERI> query = context.ERI.Where(c => c.IDClass == 5);

List<ERI> transformator = query.ToList();

return transformator;

}

public List<ERI> GetListOfOptoelectr()

{

IQueryable<ERI> query = context.ERI.Where(c => c.IDClass == 6);

List<ERI> optoelectr = query.ToList();

return optoelectr;

}

}

}

using System;

namespace Reliability

{

/// <summary>

/// Элемент системы

/// </summary>

public class Element

{

/// <summary>

/// Наименование элемента

/// </summary>

public string Name { get; set; }

/// <summary>

/// Интенсивность отказов элемента

/// </summary>

public double Lyambda { get; set; }

/// <summary>

/// Вероятность безотказной работы элемента

/// </summary>

public double P { get; set; }

/// <summary>

/// Логическая переменная элемента

/// </summary>

public char LogicArg { get; set; }

public Element(string name,char logicArg)

{

this.Name = name;

this.LogicArg = logicArg;

}

/// <summary>

/// Меняет имя элемента

/// </summary>

public void ChangeName(string name)

{

this.Name = name;

}

/// <summary>

/// Меняет логический параметр элемента

/// </summary>

/// <param name="logicArg"> Логический параметр</param>

public void ChangeLogicArg(char logicArg)

{

this.LogicArg = logicArg;

}

/// <summary>

/// Присваивает интенсивность отказов данному элементу и вычисляет вероятность его безотказной работы

/// </summary>

/// <param name="lymbda"> Интенсивность отказов</param>

public void SetLyambdaAndCalculateP(double lyambda)

{

this.Lyambda = lyambda\*Math.Pow(10,-6);

P = Math.Pow(Math.E,(double)(-Lyambda\*DataContainer.Time));

}

}

}

using System.Collections.Generic;

namespace Reliability

{

/// <summary>

/// Список элементов системы

/// </summary>

public class Elements

{

public List<Element> ElementsList { get; set; }

public Elements()

{

ElementsList = new List<Element>();

}

public Elements(List<Element> ElList)

{

ElementsList = new List<Element>(ElList);

}

public Elements(Elements elements)

{

ElementsList = new List<Element>();

foreach (Element element in elements.ElementsList)

{

ElementsList.Add(element);

}

}

}

}

using OfficeOpenXml;

using System;

using System.IO;

using OfficeOpenXml.Style;

using OfficeOpenXml.Table;

using System.Drawing;

using System.Diagnostics;

namespace Reliability

{

/// <summary>

/// Формирование отчета в excel

/// </summary>

class ExcelReport

{

public readonly string PATH = $"{Environment.CurrentDirectory}\\Отчет.xlsx";

public ReliabilityResults results { get;}

public ExcelReport(ReliabilityResults results)

{

this.results = results;

ExcelPackage.LicenseContext = LicenseContext.NonCommercial;

}

/// <summary>

/// Сохраняет отчет в папку с exe файлом

/// </summary>

public void SaveReport()

{

try

{

if (File.Exists(PATH)) File.Delete(PATH);

var file = new FileInfo(PATH);

using (var package = new ExcelPackage(file))

{

var sheet = package.Workbook.Worksheets.Add("My Sheet");

using (ExcelRange rng = sheet.Cells[2, 2, 4, 7])

{

rng.Value = "Результаты расчета надежности представлены ниже";

rng.Merge = true;

rng.Style.Font.Size = 16;

rng.Style.VerticalAlignment = OfficeOpenXml.Style.ExcelVerticalAlignment.Center;

rng.Style.HorizontalAlignment = OfficeOpenXml.Style.ExcelHorizontalAlignment.Center;

rng.Style.Border.Top.Style = ExcelBorderStyle.Medium;

rng.Style.Border.Bottom.Style = ExcelBorderStyle.Medium;

rng.Style.Border.Right.Style = ExcelBorderStyle.Medium;

rng.Style.Border.Left.Style = ExcelBorderStyle.Medium;

rng.Style.Font.Bold = true;

rng.Style.Font.Name = "Arial";

rng.Style.Font.Italic = true;

rng.Style.Fill.PatternType = OfficeOpenXml.Style.ExcelFillStyle.DarkGrid;

rng.Style.Fill.BackgroundColor.SetColor(Color.Yellow);

}

using (ExcelRange Rng = sheet.Cells["D6:E10"])

{

ExcelTable table = sheet.Tables.Add(Rng, "Reliability");

Rng.Style.Font.Bold = true;

Rng.Style.Font.Name = "Arial";

Rng.Style.Font.Italic = true;

Rng.Style.Font.Size = 12;

Rng.Style.VerticalAlignment = OfficeOpenXml.Style.ExcelVerticalAlignment.Center;

Rng.Style.HorizontalAlignment = OfficeOpenXml.Style.ExcelHorizontalAlignment.Center;

Rng.Style.Border.Top.Style = ExcelBorderStyle.Medium;

Rng.Style.Border.Bottom.Style = ExcelBorderStyle.Medium;

Rng.Style.Border.Right.Style = ExcelBorderStyle.Medium;

Rng.Style.Border.Left.Style = ExcelBorderStyle.Medium;

table.TableStyle = TableStyles.Light10;

table.Columns[0].Name = "Наименование параметра";

table.Columns[1].Name = "Значение параметра";

table.ShowFilter = false;

}

using (ExcelRange rng = sheet.Cells["D7"])

{

rng.Style.Font.Bold = false;

rng.Style.HorizontalAlignment = OfficeOpenXml.Style.ExcelHorizontalAlignment.Fill;

rng.Value = "Вероятность безотказной работы";

}

using (ExcelRange rng = sheet.Cells["D8"])

{

rng.Style.Font.Bold = false;

rng.Style.HorizontalAlignment = OfficeOpenXml.Style.ExcelHorizontalAlignment.Fill;

rng.Value = "Интенсивность отказов [1/ч]";

}

using (ExcelRange rng = sheet.Cells["D9"])

{

rng.Style.Font.Bold = false;

rng.Style.HorizontalAlignment = OfficeOpenXml.Style.ExcelHorizontalAlignment.Fill;

rng.Value = "Средняя наработка до отказа [ч]";

}

using (ExcelRange rng = sheet.Cells["D10"])

{

rng.Style.Font.Bold = false;

rng.Style.HorizontalAlignment = OfficeOpenXml.Style.ExcelHorizontalAlignment.Fill;

rng.Value = "Гамма-процентная наработка до отказа [ч]";

}

using (ExcelRange rng = sheet.Cells["E7"])

{

rng.Style.Font.Bold = false;

rng.Style.HorizontalAlignment = OfficeOpenXml.Style.ExcelHorizontalAlignment.Center;

rng.Value = $"{results.P:0.000000}";

}

using (ExcelRange rng = sheet.Cells["E8"])

{

rng.Style.Font.Bold = false;

rng.Style.HorizontalAlignment = OfficeOpenXml.Style.ExcelHorizontalAlignment.Center;

rng.Value = $"{results.Lymbda:0.000000000}";

}

using (ExcelRange rng = sheet.Cells["E9"])

{

rng.Style.Font.Bold = false;

rng.Style.HorizontalAlignment = OfficeOpenXml.Style.ExcelHorizontalAlignment.Center;

rng.Value = $"{results.Tsr:0.}";

}

using (ExcelRange rng = sheet.Cells["E10"])

{

rng.Style.Font.Bold = false;

rng.Style.HorizontalAlignment = OfficeOpenXml.Style.ExcelHorizontalAlignment.Center;

rng.Value = $"{results.Ty:0.}";

}

sheet.Cells[sheet.Dimension.Address].AutoFitColumns();

package.Save();

Process.Start(PATH);

}

}

catch { }

}

}

}

namespace Reliability

{

class ReliabilityResults

{

public double Lymbda { get; private set; }

public double Ty { get; private set; }

/// <summary>

/// Среднее время наработки до отказа

/// </summary>

public double Tsr { get; private set; }

/// <summary>

/// Вероятность безотказной работы системы

/// </summary>

public double P { get; private set; }

public ReliabilityResults(double Ty, double Tsr, double P, double lymbda)

{

this.Ty = Ty;

this.Tsr = Tsr;

this.P = P;

this.Lymbda = lymbda;

}

}

}

namespace Reliability

{

static class ReliabilityResultsContainer

{

public static ReliabilityResults Results { get; set; }

}

}

namespace Reliability

{

/// <summary>

/// Данный класс хранит промежуточные результаты расчетов

/// </summary>

static class DataContainer

{

/// <summary>

/// Логическая функция работоспособности

/// </summary>

public static string LogicFuntion { get; set; }

/// <summary>

/// Совершенная дизъюнктивная нормальная форма

/// </summary>

public static string PerfectForm { get; set; }

/// <summary>

/// Алгебраический вид СДНФ

/// </summary>

public static string AlgebraicPerfectForm { get; set; }

/// <summary>

/// Расчетная формула

/// </summary>

public static string FinalAlgebraicFormula { get; set; }

/// <summary>

/// Наименование системы

/// </summary>

public static string ProjectName { get; set; }

/// <summary>

/// Планируемое время работы системы

/// </summary>

public static int Time { get; set; } = 20000;

private static int gamma = 90;

public static int GammaProject

{

get { return gamma; }

set

{

if (value > 0 && value < 101)

gamma = value;

if (value <= 0)

gamma = 5;

if (value > 100)

gamma = 95;

}

}

}

}

//------------------------------------------------------------------------------

// <auto-generated>

// Этот код создан по шаблону.

//

// Изменения, вносимые в этот файл вручную, могут привести к непредвиденной работе приложения.

// Изменения, вносимые в этот файл вручную, будут перезаписаны при повторном создании кода.

// </auto-generated>

//------------------------------------------------------------------------------

namespace Reliability

{

using System;

using System.Data.Entity;

using System.Data.Entity.Infrastructure;

public partial class ReliabilityContext : DbContext

{

public ReliabilityContext()

: base("name=ReliabilityContext")

{

}

protected override void OnModelCreating(DbModelBuilder modelBuilder)

{

throw new UnintentionalCodeFirstException();

}

public virtual DbSet<Class> Class { get; set; }

public virtual DbSet<ERI> ERI { get; set; }

}

}

//------------------------------------------------------------------------------

// <auto-generated>

// Этот код создан по шаблону.

//

// Изменения, вносимые в этот файл вручную, могут привести к непредвиденной работе приложения.

// Изменения, вносимые в этот файл вручную, будут перезаписаны при повторном создании кода.

// </auto-generated>

//------------------------------------------------------------------------------

namespace Reliability

{

using System;

using System.Collections.Generic;

public partial class Class

{

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors")]

public Class()

{

this.ERI = new HashSet<ERI>();

}

public int ID { get; set; }

public string Name { get; set; }

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]

public virtual ICollection<ERI> ERI { get; set; }

}

}

//------------------------------------------------------------------------------

// <auto-generated>

// Этот код создан по шаблону.

//

// Изменения, вносимые в этот файл вручную, могут привести к непредвиденной работе приложения.

// Изменения, вносимые в этот файл вручную, будут перезаписаны при повторном создании кода.

// </auto-generated>

//------------------------------------------------------------------------------

namespace Reliability

{

using System;

using System.Collections.Generic;

public partial class ERI

{

public int ID { get; set; }

public string Name { get; set; }

public double Lyambda { get; set; }

public int IDClass { get; set; }

public virtual Class Class { get; set; }

}

}

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace WindowsFormsApp1

{

public partial class InputBox : Form

{

private System.Windows.Forms.Button btOk;

private Button Cancel\_Button;

private System.Windows.Forms.TextBox tbText;

private System.Windows.Forms.Label lbText;

public InputBox()

{

InitializeComponent();

this.ControlBox = false;

this.btOk = new System.Windows.Forms.Button();

this.Cancel\_Button = new Button();

this.tbText = new System.Windows.Forms.TextBox();

this.lbText = new System.Windows.Forms.Label();

this.SuspendLayout();

//

// btOk

//

this.btOk.Location = new System.Drawing.Point(45, 60);

this.btOk.Name = "btOk";

this.btOk.Size = new System.Drawing.Size(90, 30);

this.btOk.BackColor = Color.DarkSeaGreen;

this.btOk.ForeColor = Color.White;

this.btOk.Font = new Font("Tahoma", 12f,FontStyle.Regular);

this.btOk.TabIndex = 0;

this.btOk.Text = "OK";

this.btOk.Click += new System.EventHandler(this.btOk\_Click);

//

// CancelButton

//

this.Cancel\_Button.Location = new System.Drawing.Point(200, 60);

this.Cancel\_Button.Name = "CancelButton";

this.Cancel\_Button.Size = new System.Drawing.Size(90, 30);

this.Cancel\_Button.TabIndex = 0;

this.Cancel\_Button.ForeColor = Color.White;

this.Cancel\_Button.Font = new Font("Tahoma", 12f, FontStyle.Regular);

this.Cancel\_Button.Text = "Отмена";

this.Cancel\_Button.BackColor = Color.IndianRed;

this.Cancel\_Button.Click += (s, e) => { DialogResult = DialogResult.Cancel; };

//

// tbText

//

this.tbText.Location = new System.Drawing.Point(12, 25);

this.tbText.Name = "tbText";

this.tbText.Size = new System.Drawing.Size(306, 20);

this.tbText.TabIndex = 0;

this.tbText.KeyDown += new System.Windows.Forms.KeyEventHandler(this.tbText\_KeyDown);

//

// lbText

//

this.lbText.AutoSize = true;

this.lbText.Location = new System.Drawing.Point(12, 9);

this.lbText.Name = "lbText";

this.lbText.Size = new System.Drawing.Size(35, 13);

this.lbText.TabIndex = 2;

this.lbText.Text = "label1";

//

// InputBox

//

this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);

this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;

this.ClientSize = new System.Drawing.Size(330, 97);

this.Controls.Add(this.lbText);

this.Controls.Add(this.tbText);

this.Controls.Add(this.btOk);

this.Controls.Add(this.Cancel\_Button);

this.FormBorderStyle = System.Windows.Forms.FormBorderStyle.FixedToolWindow;

this.Name = "InputBox";

this.ShowIcon = false;

this.ShowInTaskbar = false;

this.StartPosition = System.Windows.Forms.FormStartPosition.CenterScreen;

this.Text = "InputBox";

this.ResumeLayout(false);

this.PerformLayout();

}

private void btOk\_Click(object sender, EventArgs e)

{

DialogResult = System.Windows.Forms.DialogResult.OK;

}

private void tbText\_KeyDown(object sender, KeyEventArgs e)

{

if (e.KeyCode == Keys.Enter)

DialogResult = System.Windows.Forms.DialogResult.OK;

if (e.KeyCode == Keys.Escape)

DialogResult = System.Windows.Forms.DialogResult.Cancel;

}

public static string ShowDialog(string label, string caption = null, string warning = null)

{

var form = new InputBox();

form.Text = caption ?? label;

form.lbText.Text = label;

var res = form.ShowDialog();

if (res != DialogResult.OK) return null;

while (string.IsNullOrWhiteSpace(form.tbText.Text) && res == DialogResult.OK)

{

MessageBox.Show(warning, "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);

res = form.ShowDialog();

}

if (res != DialogResult.Cancel) return form.tbText.Text;

else return null;

}

}

}

using System;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

namespace Reliability

{

public partial class LogicFunctionEditor : Form

{

int position;

string enabledParams;

Elements listOfElements;

private int selectedPosition//позиция курсора в текстбоксе

{

get

{

return position;

}

set

{

if (value < 0)

position = 0;

else position = value;

}

}

public LogicFunctionEditor(Elements elements)

{

InitializeComponent();

listOfElements = new Elements(elements);

enabledParams = EnableParametrs();

foreach (var control in tableLayoutPanel1.Controls.OfType<Button>())

{

if (char.IsLetter(control.Text[0]) && control.Text.Length == 1 && control.Text[0]!='v')

{

if(enabledParams.Contains(control.Text[0]))

{

control.BackColor = Color.DarkOliveGreen;

control.Enabled = true;

}

else

{

control.BackColor = enabledParams.Contains(control.Text[0]) ? Color.DarkOliveGreen : Color.IndianRed;

control.Enabled = false;

}

}

control.Click += new EventHandler(ToolButtonsClick);

}

}

private string EnableParametrs()

{

StringBuilder usingParams = new StringBuilder();

foreach (Element element in listOfElements.ElementsList)

{

usingParams.Append(element.LogicArg);

}

return usingParams.ToString();

}

private void ToolButtonsClick(object sender, EventArgs e)

{

textBox1.Focus();

Button button = sender as Button;

if ((button.Text.Length == 1 && char.IsLetter(button.Text[0]) && button.Text[0] != 'v'

|| button.Text[0] == '('))

{

textBox1.Text = textBox1.Text.Insert(selectedPosition, button.Text);

if(char.IsLetter(button.Text[0]))

{

button.Enabled = false;

button.BackColor = Color.IndianRed;

}

selectedPosition++;

}

switch (button.Text)

{

case ")":

if (selectedPosition > 0 && textBox1.Text.Where(n => n == ')').Count() < textBox1.Text.Where(n => n == '(').Count())

{

if (textBox1.Text[selectedPosition - 1] == '(')

MessageBox.Show("Выражение в скобках не может быть пустым!", "Предупреждение", MessageBoxButtons.OK, MessageBoxIcon.Warning);

else

{

textBox1.Text = textBox1.Text.Insert(selectedPosition, button.Text);

selectedPosition++;

}

}

break;

case "Back":

if (selectedPosition > 0)

{

if (selectedPosition < textBox1.Text.Length && (textBox1.Text[selectedPosition] == '^' || textBox1.Text[selectedPosition] == 'v')

&& char.IsLetter(textBox1.Text[selectedPosition - 1]))

{

textBox1.Text = textBox1.Text.Remove(textBox1.SelectionStart - 1, 2);

selectedPosition -= 2;

}

else

{

if (textBox1.SelectionStart - 1 < 0)

return;

textBox1.Text = textBox1.Text.Remove(textBox1.SelectionStart - 1, 1);

selectedPosition--;

}

}

break;

case "Clear":

{

textBox1.Clear();

foreach (Button but in tableLayoutPanel1.Controls)

{

if (but.Text.Length == 1 && enabledParams.Contains(but.Text[0]) && !but.Enabled)

{

but.Enabled = true;

but.BackColor = Color.DarkOliveGreen;

}

}

selectedPosition = 0;

break;

}

case "->":

{

if (selectedPosition < textBox1.Text.Length)

{

selectedPosition++;

textBox1.SelectionStart = selectedPosition;

}

break;

}

case "<-":

{

if (selectedPosition != 0)

{

selectedPosition--;

textBox1.SelectionStart = selectedPosition;

}

break;

}

case "^":

{

if (selectedPosition == textBox1.Text.Length && selectedPosition != 0 &&

(char.IsLetter(textBox1.Text[selectedPosition - 1]) || (textBox1.Text[selectedPosition - 1] == ')'))

&& textBox1.Text[selectedPosition - 1] != 'v')

{

textBox1.Text = textBox1.Text.Insert(selectedPosition, button.Text);

selectedPosition++;

}

if (selectedPosition < textBox1.Text.Length && selectedPosition > 0

&& char.IsLetter(textBox1.Text[selectedPosition - 1]) && char.IsLetter(textBox1.Text[selectedPosition]))

{

textBox1.Text = textBox1.Text.Insert(selectedPosition, button.Text);

selectedPosition++;

}

break;

}

case "v":

{

if (selectedPosition == textBox1.Text.Length && selectedPosition != 0

&& (char.IsLetter(textBox1.Text[selectedPosition - 1]) || (textBox1.Text[selectedPosition - 1] == ')'))

&& textBox1.Text[selectedPosition - 1] != 'v')

{

textBox1.Text = textBox1.Text.Insert(selectedPosition, button.Text);

selectedPosition++;

}

if (selectedPosition < textBox1.Text.Length && selectedPosition > 0

&& char.IsLetter(textBox1.Text[selectedPosition - 1]) && char.IsLetter(textBox1.Text[selectedPosition]))

{

textBox1.Text = textBox1.Text.Insert(selectedPosition, button.Text);

selectedPosition++;

}

break;

}

}

CheckOperations();

textBox1.SelectionStart = selectedPosition;

}

private void CheckOperations()

{

textBox1.Focus();

string result = textBox1.Text;

for (int i = 0; i < result.Length - 1; i++)

{

if (result[i] == ')' && char.IsLetter(result[i + 1]) && result[i + 1] != 'v')

{

result = result.Insert(i + 1, "^");

selectedPosition++;

}

if (char.IsLetter(result[i]) && result[i] != 'v' && result[i + 1] != 'v' && (result[i + 1] == '(' || char.IsLetter(result[i + 1])))

{

result = result.Insert(i + 1, "^");

selectedPosition++;

}

if (result[i] == ')' && result[i + 1] == '(')

{

result = result.Insert(i + 1, "^");

selectedPosition++;

}

}

textBox1.Text = result;

}

private void textBox1\_MouseEnter(object sender, EventArgs e)

{

textBox1.Enabled = false;

}

private void textBox1\_MouseLeave(object sender, EventArgs e)

{

textBox1.Enabled = true;

}

private void LogicFunctionEditor\_Load(object sender, EventArgs e)

{

position = 0;

}

private void materialButton1\_Click(object sender, EventArgs e)

{

// пытаемся получить СДНФ, тем самым проверяем корректность ввода логической функции

try

{

DataContainer.PerfectForm = SDNF.GetSDNF(textBox1.Text); //В случае успеха при построение СДНФ, сохраняем ее

DataContainer.AlgebraicPerfectForm = SDNF.Convert\_to\_expr(DataContainer.PerfectForm); //Сохраняем алгебраическую форму

DataContainer.LogicFuntion = textBox1.Text; // сохраняем логическую функцию

LogicalMethod method = new LogicalMethod();

method.ReliabilityCalculate(listOfElements,DataContainer.AlgebraicPerfectForm);

DialogResult = DialogResult.OK;

}

catch(System.InvalidOperationException)

{

MessageBox.Show("Ошибка в построение логической функции!", "Предупреждение", MessageBoxButtons.OK, MessageBoxIcon.Warning);

}

}

private void materialButton2\_Click(object sender, EventArgs e)

{

DialogResult = DialogResult.Cancel;

}

}

}

using Reliability;

using System;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

namespace WindowsFormsApp1

{

public partial class MainForm : Form

{

Project openProject;

bool removed;

string logicParametrs = "ABCDEFGHIJKLMNOPQRSTUW";

Elements currentElements;

DataGridView projectView = new DataGridView();

DataGridView elementsView = new DataGridView();

public MainForm()

{

InitializeComponent();

currentElements = new Elements();

toolStripSplitButton1.Click += new EventHandler(CreateNewProject);

OpenProjectStripButton.Click += new EventHandler(OpenProjectToolStripMenuItem\_Click);

SaveProjectStripButton.Click += new EventHandler(SaveProjectToolStripMenuItem\_Click);

SaveAsStripButton.Click += new EventHandler(SaveAsToolStripMenuItem\_Click);

CreateNewToolStripMenuItem.Click += new EventHandler(CreateNewProject);

}

private void RestoreLogicParams()

{

logicParametrs = "ABCDEFGHIJKLMNOPQRSTUW";

}

private void ProjectViewCellValueChanged(object sender, DataGridViewCellEventArgs e)

{

if (projectView.SelectedCells.Contains(projectView[1, 0]))

{

string newName = projectView[1, 0].Value.ToString();

DataContainer.ProjectName = newName;

treeView1.TopNode.Text = newName;

}

else if (projectView.SelectedCells.Contains(projectView[1, 1]))

{

int time = int.Parse(projectView[1, 1].Value.ToString());

DataContainer.Time = time;

}

else if (projectView.SelectedCells.Contains(projectView[1, 2]))

{

int gamma = int.Parse(projectView[1, 2].Value.ToString());

DataContainer.GammaProject = gamma;

projectView[1, 2].Value = DataContainer.GammaProject;

}

}

private void ElementViewCellValueChanged(object sender, DataGridViewCellEventArgs e)

{

int index = treeView1.SelectedNode.Index;

// При изменении названия элемента

if (elementsView.SelectedCells.Contains(elementsView[1, 0]))

{

currentElements.ElementsList[index].ChangeName(elementsView[1, 0].Value.ToString());

treeView1.SelectedNode.Text = $"{elementsView[1, 0].Value} ({currentElements.ElementsList[index].LogicArg})";

}

else if (elementsView.SelectedCells.Contains(elementsView[1, 1]))

{

double value;

if (double.TryParse(elementsView[1, 1].Value.ToString(), out value))

{

currentElements.ElementsList[index].SetLyambdaAndCalculateP(value);

elementsView[1, 2].Value = currentElements.ElementsList[index].P;

}

}

}

private void CellValidating(object sender, DataGridViewCellValidatingEventArgs e)

{

int newInteger;

var dataGrid = sender as DataGridView;

// Проверка строкового формата

if (dataGrid.SelectedCells.Contains(dataGrid[1, 0]))

{

if (string.IsNullOrWhiteSpace(e.FormattedValue.ToString()))

{

e.Cancel = true;

MessageBox.Show("Данная строка не может быть пустой!", "Ошибка ввода", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

// Проверка полей со временем и гамма-процентом в гриде с проектом

if (dataGrid != elementsView && (dataGrid.SelectedCells.Contains(dataGrid[1, 1]) || dataGrid.SelectedCells.Contains(dataGrid[1, 2])))

{

if (!int.TryParse(e.FormattedValue.ToString(),

out newInteger) || newInteger < 0)

{

e.Cancel = true;

MessageBox.Show("Допускается ввод только положительных целочисленных значений!", "Ошибка ввода", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

// Проверка полей с интенсивностью и вероятностью в гриде с элементами

if (dataGrid != projectView && dataGrid.SelectedCells.Contains(dataGrid[1, 1]))

{

double newValue;

if (!Double.TryParse(e.FormattedValue.ToString(),

out newValue) || newValue < 0)

{

e.Cancel = true;

MessageBox.Show("Допускается ввод только положительных значений!", "Ошибка ввода", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

}

}

private void MainForm\_Load(object sender, EventArgs e)

{

RestoreLogicParams();

// Динамически создаем ДатаГриды для редактирования данных

projectView.Dock = elementsView.Dock = DockStyle.Fill;

projectView.AutoSizeRowsMode = elementsView.AutoSizeRowsMode = DataGridViewAutoSizeRowsMode.None;

projectView.RowTemplate.Height = elementsView.RowTemplate.Height = 30;

projectView.MultiSelect = elementsView.MultiSelect = false;

projectView.AllowUserToAddRows = elementsView.AllowUserToAddRows = false;

projectView.AllowUserToDeleteRows = elementsView.AllowUserToDeleteRows = false;

projectView.AllowUserToOrderColumns = elementsView.AllowUserToOrderColumns = false;

projectView.AllowUserToResizeRows = elementsView.AllowUserToResizeRows = false;

projectView.AllowUserToResizeColumns = elementsView.AllowUserToResizeColumns = false;

projectView.AutoSizeColumnsMode = elementsView.AutoSizeColumnsMode = DataGridViewAutoSizeColumnsMode.Fill;

projectView.RowHeadersVisible = elementsView.RowHeadersVisible = false;

projectView.BackgroundColor = elementsView.BackgroundColor = SystemColors.MenuBar;

projectView.CellBorderStyle = elementsView.CellBorderStyle = DataGridViewCellBorderStyle.Single;

projectView.ColumnHeadersHeightSizeMode = elementsView.ColumnHeadersHeightSizeMode = DataGridViewColumnHeadersHeightSizeMode.AutoSize;

projectView.Columns.Add("PropertyName", "Наименование");

projectView.Columns[0].ReadOnly = true;

projectView.Columns.Add("PropertyValue", "Значение");

projectView.Rows.Add("Наименование системы");

projectView.Rows.Add("Планируемое время работы [ч]", DataContainer.Time);

projectView.Rows.Add("Гамма процент [%]", DataContainer.GammaProject);

elementsView.Columns.Add("PropertyName", "Наименование");

elementsView.Columns[0].ReadOnly = true;

elementsView.Columns.Add("PropertyValue", "Значение");

elementsView.Rows.Add("Наименование элемента [1/ч]");

elementsView.Rows.Add("Интенсивность отказов (10)" + '\u2076' + " [1/ч]");

elementsView.Rows.Add("Вероятность безотказной работы");

elementsView[1, 2].ReadOnly = true;

projectView.Columns[0].SortMode = elementsView.Columns[0].SortMode = DataGridViewColumnSortMode.NotSortable;

projectView.Columns[1].SortMode = elementsView.Columns[1].SortMode = DataGridViewColumnSortMode.NotSortable;

// Подписываем гриды на события

projectView.CellValidating += new DataGridViewCellValidatingEventHandler(CellValidating);

projectView.CellValueChanged += new DataGridViewCellEventHandler(ProjectViewCellValueChanged);

elementsView.CellValidating += new DataGridViewCellValidatingEventHandler(CellValidating);

elementsView.CellValueChanged += new DataGridViewCellEventHandler(ElementViewCellValueChanged);

dataGridView1.Rows.Add("Вероятность безотказной работы системы");

dataGridView1.Rows.Add("Интенсивность отказов системы [1/ч]");

dataGridView1.Rows.Add("Средняя наработка до отказа [ч]");

dataGridView1.Rows.Add("Гамма-процентная наработка до отказа [ч]");

dataGridView1.Font = new Font("Tahoma", 12);

HideTabs();

}

private void HideTabs()

{

tabControl2.TabPages.Remove(StepByStepPage); tabControl2.TabPages.Remove(ReliabilityTab);

}

private void treeView1\_MouseUp(object sender, MouseEventArgs e)

{

if (e.Button == MouseButtons.Right)

{

try

{

удалитьВыбранныйКомпонентToolStripMenuItem.Enabled = treeView1.SelectedNode == treeView1.TopNode ? false : true;

добавитьКомпонентToolStripMenuItem.Enabled = treeView1.SelectedNode == treeView1.TopNode ? true : false;

модульToolStripMenuItem.Enabled = treeView1.SelectedNode != treeView1.TopNode ? false : true;

contextMenuStrip1.Show(MousePosition, ToolStripDropDownDirection.Right);

}

catch (Exception ex)

{

MessageBox.Show(ex.Message);

}

}

}

/// <summary>

/// Добавляет новый элемент в проект

/// </summary>

private void модульToolStripMenuItem\_Click(object sender, EventArgs e)

{

string name = InputBox.ShowDialog("Введите наименование элемента: ", "Добавление элемента", "Необходимо ввести наименование элемента системы! ");

if (name != null)

{

if (treeView1.SelectedNode.Level == 0 && logicParametrs.Length > 0)

{

treeView1.Nodes[0].Nodes.Add($"{name} ({logicParametrs[0]})");

currentElements.ElementsList.Add(new Element(name, logicParametrs[0]));

currentElements.ElementsList = currentElements.ElementsList.OrderBy(n => n.LogicArg).ToList();

logicParametrs = logicParametrs.Remove(0, 1);

}

}

}

private void удалитьВыбранныйКомпонентToolStripMenuItem\_Click(object sender, EventArgs e)

{

removed = true;

currentElements.ElementsList.RemoveAt(treeView1.SelectedNode.Index);

treeView1.SelectedNode.Remove();

}

/// <summary>

/// Переприсваивает всем элементам логическую переменную в алфавитном порядке

/// </summary>

private void SortElements()

{

RestoreLogicParams();

int count = currentElements.ElementsList.Count;

for (int i = 0; i < count; i++)

{

currentElements.ElementsList[i].ChangeLogicArg(logicParametrs[0]);

treeView1.TopNode.Nodes[i].Text = $"{currentElements.ElementsList[i].Name} ({currentElements.ElementsList[i].LogicArg})";

logicParametrs = logicParametrs.Remove(0, 1);

}

}

private void CreateNewProject(object sender, EventArgs e)

{

DialogResult dialog = DialogResult.OK;

if (treeView1.Nodes.Count > 0)

{

dialog = MessageBox.Show("Текущий проект будет закрыт. Все несохранненные данные будут утеряны. " +

"Вы уверены, что хотите продолжить?", "Предупреждение", MessageBoxButtons.OKCancel, MessageBoxIcon.Warning);

if (dialog == DialogResult.OK)

{

treeView1.Nodes.Clear();

PropertyTab.Controls.Clear();

ReliabilityTab.Controls.Clear();

StepByStepPage.Controls[0].Text = String.Empty;

}

}

if (dialog == DialogResult.OK)

{

string name = InputBox.ShowDialog("Введите наименование системы", "Создание проекта", "Необходимо ввести наименование проекта!");

if (string.IsNullOrWhiteSpace(name)) return;

HideTabs();

currentElements = new Elements();

PropertyTab.Controls.Add(projectView);

DataContainer.ProjectName = name;

projectView[1, 0].Value = name;

treeView1.Enabled = true;

treeView1.Nodes.Add(name);

treeView1.TopNode.BackColor = Color.Green;

treeView1.TopNode.ForeColor = Color.White;

RestoreLogicParams();

}

}

private void treeView1\_NodeMouseClick(object sender, TreeNodeMouseClickEventArgs e)

{

if (e.Button == MouseButtons.Right)

{

treeView1.SelectedNode = e.Node;

}

}

/// <summary>

/// Расчет надежности

/// </summary>

private void CalculateReliabilityButton\_Click(object sender, EventArgs e)

{

if (currentElements.ElementsList.Count < 1)

{

MessageBox.Show("Перед расчетом необходимо добавить элементы в систему!", "Отсутствуют элементы для расчета", MessageBoxButtons.OK, MessageBoxIcon.Information);

return;

}

LogicFunctionEditor editor = new LogicFunctionEditor(currentElements);

if (editor.ShowDialog() == DialogResult.OK)

{

if (!tabControl2.TabPages.Contains(ReliabilityTab))

tabControl2.TabPages.Add(ReliabilityTab);

if (!tabControl2.TabPages.Contains(StepByStepPage))

tabControl2.TabPages.Add(StepByStepPage);

materialMultiLineTextBox1.Clear();

UpdateReliabilityCalculations();

сформироватьОтчетToolStripMenuItem.Enabled = true;

}

}

private void UpdateReliabilityCalculations()

{

StringBuilder builder = new StringBuilder();

dataGridView1[1, 0].Value = ReliabilityResultsContainer.Results.P;

dataGridView1[1, 1].Value = ReliabilityResultsContainer.Results.Lymbda;

dataGridView1[1, 2].Value = ReliabilityResultsContainer.Results.Tsr;

dataGridView1[1, 3].Value = ReliabilityResultsContainer.Results.Ty;

builder.AppendLine("Исходная логическая функция работоспособности:");

builder.AppendLine(DataContainer.LogicFuntion);

builder.AppendLine("Преобразуем логическую функцию в совершенную нормальную дизьюнктивную форму: ");

builder.AppendLine(DataContainer.PerfectForm);

builder.AppendLine("Заменяем логические операции на алгебраические: ");

builder.AppendLine(DataContainer.AlgebraicPerfectForm);

builder.AppendLine("Проведем замену логических переменных на соответствующие вероятности безотказной работы. В итоге, получим расчетную формулу: ");

builder.AppendLine(DataContainer.FinalAlgebraicFormula);

builder.AppendLine("\r\nВ результате, получили следующие показатели надежности системы: ");

builder.AppendLine($"Вероятность безотказной работы P({DataContainer.Time}): {ReliabilityResultsContainer.Results.P:0.000000}");

builder.AppendLine($"Среднее время наработки до отказа: {ReliabilityResultsContainer.Results.Tsr:0.}");

builder.AppendLine($"Интенсивность отказов: {ReliabilityResultsContainer.Results.Lymbda}");

builder.AppendLine($"Гамма-процентная наработка до отказа T({DataContainer.GammaProject}) = {ReliabilityResultsContainer.Results.Ty:0.}");

materialMultiLineTextBox1.Text = builder.ToString();

}

private void treeView1\_AfterSelect(object sender, TreeViewEventArgs e)

{

PropertyTab.Controls.Clear();

if (treeView1.SelectedNode.Level == 0)

{

try

{

projectView.CurrentCell = null;

}

catch { }

PropertyTab.Text = "Свойства проекта";

PropertyTab.Controls.Add(projectView);

}

else

{

try

{

elementsView.CurrentCell = null;

}

catch { }

if (removed) //Если элемент удалялся, заново перестроить дерево

{

SortElements();

removed = false;

}

PropertyTab.Text = "Свойства элемента";

PropertyTab.Controls.Add(elementsView);

int index = e.Node.Index;

elementsView[1, 1].Value = currentElements.ElementsList[index].Lyambda == 0 ? string.Empty : currentElements.ElementsList[index].Lyambda.ToString();

elementsView[1, 2].Value = currentElements.ElementsList[index].P == 0 ? string.Empty : currentElements.ElementsList[index].P.ToString();

elementsView[1, 0].Value = currentElements.ElementsList[index].Name;

}

}

private void оПрограммеToolStripMenuItem\_Click(object sender, EventArgs e)

{

MessageBox.Show("Программа предназначена для расчета надежности сложных технических систем. " +

"Разработал Хабибуллин Александр Эдуардович, студент группы 4413, КНИТУ-КАИ.", "О программе", MessageBoxButtons.OK, MessageBoxIcon.Information);

}

// предусмотреть добавление логической переменной в текст элемента

private void OpenProjectToolStripMenuItem\_Click(object sender, EventArgs e)

{

DialogResult dialogResult = DialogResult.OK;

if (treeView1.Nodes.Count > 0)

{

dialogResult = MessageBox.Show("Текущий проект будет закрыт. Все несохранненные данные будут утеряны. " +

"Вы уверены, что хотите продолжить?", "Предупреждение", MessageBoxButtons.OKCancel, MessageBoxIcon.Warning);

}

if (dialogResult == DialogResult.OK)

{

openProject = Project.OpenProject();

if (openProject != null)

{

RestoreLogicParams();

toolStripTextBox1.Text = openProject.Path;

currentElements = openProject.ProjectElemets;

if (treeView1.Nodes.Count > 0) treeView1.Nodes.Clear();

treeView1.Nodes.Add(openProject.Name);

if (currentElements.ElementsList.Count > 0)

{

foreach (Element element in currentElements.ElementsList)

{

treeView1.TopNode.Nodes.Add(element.Name + " (" + element.LogicArg + ")");

logicParametrs = logicParametrs.Remove(logicParametrs.IndexOf(element.LogicArg), 1);

}

}

treeView1.Enabled = true;

treeView1.TopNode.BackColor = Color.Green;

treeView1.TopNode.ForeColor = Color.White;

projectView[1, 0].Value = DataContainer.ProjectName = openProject.Name;

projectView[1, 1].Value = DataContainer.Time = openProject.Time;

projectView[1, 2].Value = DataContainer.GammaProject = openProject.Gamma;

}

}

}

private void SaveProjectToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (currentElements.ElementsList.Count > 0)

{

Project project = openProject ?? new Project(currentElements);

project.SaveProject();

toolStripTextBox1.Text = project.Path;

}

else MessageBox.Show("Проект пустой. Добавьте хотя бы один элемент!", "Ошибка при сохранении проекта", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

private void SaveAsToolStripMenuItem\_Click(object sender, EventArgs e)

{

if (currentElements.ElementsList.Count > 0)

{

Project project = new Project(currentElements);

project.SaveProjectAs();

toolStripTextBox1.Text = project.Path;

}

else MessageBox.Show("Проект пустой. Добавьте хотя бы один элемент!", "Ошибка при сохранении проекта", MessageBoxButtons.OK, MessageBoxIcon.Error);

}

private void сформироватьОтчетToolStripMenuItem\_Click(object sender, EventArgs e)

{

ExcelReport report = new ExcelReport(ReliabilityResultsContainer.Results);

report.SaveReport();

сформироватьОтчетToolStripMenuItem.Enabled = false;

}

private void ERIMenuItem\_Click(object sender, EventArgs e)

{

DataBaseForm form = new DataBaseForm();

if (form.ShowDialog() == DialogResult.OK)

{

if (treeView1.SelectedNode.Level == 0 && logicParametrs.Length > 0)

{

treeView1.Nodes[0].Nodes.Add($"{form.ERIName} ({logicParametrs[0]})");

currentElements.ElementsList.Add(new Element(form.ERIName, logicParametrs[0]));

currentElements.ElementsList[currentElements.ElementsList.Count - 1].SetLyambdaAndCalculateP(form.lymbda);

logicParametrs = logicParametrs.Remove(0, 1);

}

}

}

}

}

using Newtonsoft.Json;

using System.IO;

using System.Windows.Forms;

namespace Reliability

{

/// <summary>

/// Текущий проект

/// </summary>

class Project

{

/// <summary>

/// Список элементов системы

/// </summary>

public Elements ProjectElemets { get; set; }

/// <summary>

/// Логическая функция работоспособности системы

/// </summary>

public string LogicFunction { get; set; }

/// <summary>

/// Путь, по которому был сохранен проект

/// </summary>

public string Path { get; set; }

public string Name { get; set; }

public int Time { get; set; }

public int Gamma { get; set; }

public Project()

{

}

public Project(Elements elements)

{

ProjectElemets = new Elements(elements);

}

/// <summary>

/// Сохраняет проект

/// </summary>

public void SaveProject()

{

string finalPath;

SaveFileDialog saveProject = new SaveFileDialog();

saveProject.DefaultExt = ".rlb";

saveProject.Filter = "Reliability project|\*.rlb";

if (File.Exists(Path)) finalPath = Path;

else if (saveProject.ShowDialog() == DialogResult.OK && saveProject.FileName.Length > 0) finalPath = saveProject.FileName;

else return;

using (StreamWriter sw = new StreamWriter(finalPath, false))

{

LogicFunction = LogicFunction ?? DataContainer.LogicFuntion;

Path = finalPath;

Name = DataContainer.ProjectName;

Gamma = DataContainer.GammaProject;

Time = DataContainer.Time;

string output = JsonConvert.SerializeObject(this);

sw.Write(output);

}

}

/// <summary>

/// Сохранить проект как

/// </summary>

public void SaveProjectAs()

{

SaveFileDialog saveProject = new SaveFileDialog();

saveProject.DefaultExt = ".rlb";

saveProject.Filter = "Reliability project|\*.rlb";

if (saveProject.ShowDialog() == DialogResult.OK && saveProject.FileName.Length > 0)

{

using (StreamWriter sw = new StreamWriter(saveProject.FileName, false))

{

LogicFunction = DataContainer.LogicFuntion;

Path = saveProject.FileName;

Name = DataContainer.ProjectName;

Gamma = DataContainer.GammaProject;

Time = DataContainer.Time;

string output = JsonConvert.SerializeObject(this);

sw.Write(output);

}

}

}

/// <summary>

/// Открывает проект

/// </summary>

public static Project OpenProject()

{

OpenFileDialog openProject = new OpenFileDialog();

openProject.DefaultExt = ".rlb";

openProject.Filter = "Reliability project|\*.rlb";

if (openProject.ShowDialog() == DialogResult.OK && openProject.FileName.Length > 0)

{

using (StreamReader reader = new StreamReader(openProject.FileName))

{

var filetext = reader.ReadToEnd();

return JsonConvert.DeserializeObject<Project>(filetext);

}}else return null; } }}