

Инженерная школа энергетики

09.04.03 Прикладная информатика

«Информационные технологии в электроэнергетике»

Отделение электроэнергетики и электротехники

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА МАГИСТРАНТА**

|  |
| --- |
| **Тема работы** |
| **Разработка программного обеспечения для формирования эквивалентных математических моделей солнечных электростанций на основе данных информационной модели энергосистемы** |

УДК 621.311.24:519.876:004.415

Обучающийся

|  |  |  |  |
| --- | --- | --- | --- |
| **Группа** | **ФИО** | **Подпись** | **Дата** |
| О-5КМ01 | Чернобров Михаил Евгеньевич |  |  |

Руководитель ВКР

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Должность** | **ФИО** | **Ученая степень, звание** | **Подпись** | **Дата** |
| доцент ОЭЭ ИШЭ ТПУ | Прохоров А.В. | к.т.н., доцент |  |  |

Консультант

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Должность** | **ФИО** | **Ученая степень, звание** | **Подпись** | **Дата** |
| доцент КСУП ТУСУР | Калентьев А.А. | к.т.н., доцент |  |  |

**КОНСУЛЬТАНТЫ ПО РАЗДЕЛАМ:**

По разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Должность** | **ФИО** | **Ученая степень, звание** | **Подпись** | **Дата** |
| доцент ОСГН ШБИП ТПУ | Маланина В.А. | к.э.н., доцент |  |  |

По разделу «Социальная ответственность»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Должность** | **ФИО** | **Ученая степень, звание** | **Подпись** | **Дата** |
| доцент ООД ШБИП ТПУ | Антоневич О.А. | к.б.н, доцент |  |  |

**ДОПУСТИТЬ К ЗАЩИТЕ:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Руководитель ООП/ОПОП, должность** | **ФИО** | **Ученая степень, звание** | **Подпись** | **Дата** |
| доцент ОЭЭ ИШЭ ТПУ | Прохоров А. В. | к.т.н., доцент |  |  |

Томск – 2023 г.**Компетенции выпускника основной образовательной программы магистратуры по направлению 09.04.03 «Прикладная информатика»**

| **Код компетенции** | **Наименование компетенции** |
| --- | --- |
| УК(У)-1 | Способен осуществлять критический анализ проблемных ситуаций на основе системного подхода, вырабатывать стратегию действий |
| УК(У)-2 | Способен управлять проектом на всех этапах его жизненного цикла |
| УК(У)-3 | Способен организовывать и руководить работой команды, вырабатывая командную стратегию для достижения поставленной цели |
| УК(У)-4 | Способен применять современные коммуникативные технологии, в том числе на иностранном (-ых) языке (-ах), для академического и профессионального взаимодействия |
| УК(У)-5 | Способен анализировать и учитывать разнообразие культур в процессе межкультурного взаимодействия |
| УК(У)-6 | Способен определять и реализовывать приоритеты собственной деятельности и способы ее совершенствования на основе самооценки |
| ОПК(У)-1 | Способен самостоятельно приобретать, развивать и применять математические, естественнонаучные, социально-экономические и профессиональные знания для решения нестандартных задач, в том числе в новой или незнакомой среде и в междисциплинарном контексте |
| ОПК(У)-2 | Способен разрабатывать оригинальные алгоритмы и программные средства, в том числе с использованием современных интеллектуальных технологий, для решения профессиональных задач |
| ОПК(У)-3 | Способен анализировать профессиональную информацию, выделять в ней главное, структурировать, оформлять и представлять в виде аналитических обзоров с обоснованными выводами и рекомендациями |
| ОПК(У)-4 | Способен применять на практике новые научные принципы и методы исследований |
| ОПК(У)-5 | Способен разрабатывать и модернизировать программное и аппаратное обеспечение информационных и автоматизированных систем |
| ОПК(У)-6 | Способен исследовать современные проблемы и методы прикладной информатики и развития информационного общества |
| ОПК(У)-7 | Способен использовать методы научных исследований и математического моделирования в области проектирования и управления информационными системами |
| ОПК(У)-8 | Способен осуществлять эффективное управление разработкой программных средств и проектов |
| ПК(У)-1 | Способен анализировать бизнес-процессы в электроэнергетике, создавать и применять информационные модели для их автоматизации |
| ПК(У)-2 | Способен самостоятельно осваивать и применять информационные технологии для автоматизации бизнес-процессов в электроэнергетике |
| ПК(У)-3 | Способен выявлять ошибки и неисправности в работе информационных систем, предлагать решения по их устранению, реализовывать технические мероприятия по обеспечению требований к надежности и информационной безопасности |



Инженерная школа энергетики

09.04.03 Прикладная информатика

Отделение электроэнергетики и электротехники

УТВЕРЖДАЮ:

Руководитель ООП/ОПОП

\_\_\_\_\_\_\_ 08.07.2022 Прохоров А.В. (Подпись) (Дата) (Ф.И.О.)

**ЗАДАНИЕ**

**на выполнение выпускной квалификационной работы**

Обучающийся:

|  |  |
| --- | --- |
| **Группа** | **ФИО** |
| О-5КМ01 | Чернобров Михаил Евгеньевич |

Тема работы:

|  |  |
| --- | --- |
| Разработка программного обеспечения для формирования эквивалентных математических моделей солнечных электростанций на основе данных информационной модели энергосистемы | |
| Утверждена приказом директора (дата, номер) | 23.12.2022, №357-13/с |

|  |  |
| --- | --- |
| Срок сдачи студентом выполненной работы: | 20.01.2023 |

**ТЕХНИЧЕСКОЕ ЗАДАНИЕ:**

|  |  |  |
| --- | --- | --- |
| **Исходные данные к работе**  *(наименование объекта исследования или проектирования; производительность или нагрузка; режим работы (непрерывный, периодический, циклический и т. д.); вид сырья или материал изделия; требования к продукту, изделию или процессу; особые требования к функционированию (эксплуатации) объекта или изделия в плане безопасности эксплуатации, влияния на окружающую среду, энергозатратам; экономический анализ и т. д.)* | 1. Техническое задание на разработку программного обеспечения 2. Регламент взаимодействия структурных подразделений исполнительного аппарата и филиалов АО «СО ЕЭС» ОДУ, филиалов АО «СО ЕЭС» РДУ при актуализации данных информационной модели 3. Регламент взаимодействия ОДУ Сибири и РДУ ОЗ ОДУ Сибири при формировании и внесении изменений в расчетные модели ОЭС Сибири 4. Каноническая модель ОИК СК-11 применяемая в АО «СО ЕЭС» 5. Информационная модель Ининской СЭС 6. Расчетные модели установившихся режимов ЭС Республики Алтай и Алтайского края для ПК «RastrWin3» 7. Проектная документация «Ининская солнечная электростанция мощностью 25 МВт (1 очередь 10 МВт) 8. Проектная документация «Ининская солнечная электростанция мощностью 25 МВт (2 очередь 15 МВт) 9. Данные субъекта по Приказу №102 «Информация о параметрах оборудования Ининская СЭС» | |
| **Перечень разделов пояснительной записки подлежащих исследованию, проектированию и разработке**  *(аналитический обзор литературных источников с целью выяснения достижений мировой науки техники в рассматриваемой области; постановка задачи исследования, проектирования, конструирования; содержание процедуры исследования, проектирования, конструирования; обсуждение результатов выполненной работы; наименование дополнительных разделов, подлежащих разработке; заключение по работе)* | 1. Анализ предметной области 2. Проектирование программного обеспечения 3. Программная реализация 4. Тестирование программного обеспечения 5. Разработка руководства пользователя 6. Выполнение раздела «Финансовый менеджмент» 7. Выполнение раздела «Социальная ответственность» | |
| **Перечень графического материала**  *(с точным указанием обязательных чертежей)* | – | |
| **Консультанты по разделам выпускной квалификационной работы** | | |
| **Раздел** | | **Консультант** |
| Финансовый менеджмент | | Маланина В.А., к.э.н., доцент ОСГН ШБИП ТПУ |
| Социальная ответственность | | Антоневич О.А., к.б.н, доцент ООД ШБИП ТПУ |

|  |  |
| --- | --- |
| **Дата выдачи задания на выполнение выпускной квалификационной работы по линейному графику** | 08.07.2022 |

**Задание выдал руководитель / консультант** (при наличии)**:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Должность** | **ФИО** | **Ученая степень, звание** | **Подпись** | **Дата** |
| доцент ОЭЭ ИШЭ ТПУ | Прохоров А.В. | к.т.н., доцент |  | 08.07.2022 |
| доцент КСУП ТУСУР | Калентьев А.А. | к.т.н., доцент |  | 08.07.2022 |

**Задание принял к исполнению обучающийся:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Группа** | **ФИО** | **Подпись** | **Дата** |
| О-5КМ01 | Чернобров Михаил Евгеньевич |  | 08.07.2022 |

**ЗАДАНИЕ ДЛЯ РАЗДЕЛА**

**«ФИНАНСОВЫЙ МЕНЕДЖМЕНТ, РЕСУРСОЭФФЕКТИВНОСТЬ И РЕСУРСОСБЕРЕЖЕНИЕ»**

Студенту:

|  |  |
| --- | --- |
| **Группа** | **ФИО** |
| О-5КМ01 | Черноброву Михаилу Евгеньевичу |

|  |  |  |  |
| --- | --- | --- | --- |
| **Школа** | **Инженерная школа энергетики** | **Отделение школы (НОЦ)** | **Отделение электроэнергетики и электротехники** |
| **Уровень образования** | Магистратура | **Направление/специальность** | 09.04.03 Прикладная информатика |

|  |  |  |
| --- | --- | --- |
| **Исходные данные к разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»:** | | |
| 1. *Стоимость ресурсов научного исследования (НИ): материально-технических, энергетических, финансовых, информационных и человеческих* | Стоимость материальных ресурсов - по средней стоимости по г. Томску. Оклады и премии- в соответствии с Положением об оплате труда в Томском политехническом университете. | |
| 1. *Нормы и нормативы расходования ресурсов* | Премии – 10%  Доплаты и надбавки – 5-10%;  Районный коэффициент – 30% | |
| 1. *Используемая система налогообложения, ставки налогов, отчислений, дисконтирования и кредитования* | Государственное социальное страхование – 2,9%  Пенсионный фонд – 22%  Обязательное медицинское страхование – 5,1% | |
| **Перечень вопросов, подлежащих исследованию, проектированию и разработке:** | | |
| 1. *Оценка коммерческого и инновационного потенциала НТИ* | Определение потенциальных потребителей. Проведение анализ конкурентных технических решений. SWOT‑анализ проекта. Оценка готовности и методы коммерциализации. | |
| 1. *Разработка устава научно-технического проекта* | Формирование целей и результатов проекта. Организационная структура проекта. Выявление ограничений и допущения проекта. | |
| 1. *Планирование процесса управления НТИ: структура и график проведения, бюджет, риски и организация закупок* | Построение иерархической структура проекта. Фиксация контрольных событий. Разработка плана проекта. Определение бюджета проекта. Организационная структура управления проектом, матрица ответственности, план управления коммуникациями и реестр рисков. | |
| 1. *Определение ресурсной, финансовой, экономической эффективности* | Проведение оценки экономической эффективности, ресурсоэффективности и сравнительной эффективности | |
| **Перечень графического материала** *(с точным указанием обязательных чертежей):* | | |
| 1. *Сегментирование рынка* 2. *Оценка конкурентоспособности технических решений* 3. *Матрица SWOT* 4. *Иерархическая структура проекта* 5. *План график проекта* 6. *Организационная структура проекта.* | | |
| **Дата выдачи задания для раздела по линейному графику** | | 08.07.2022 |

**Задание выдал консультант:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Должность** | **ФИО** | **Ученая степень, звание** | **Подпись** | **Дата** |
| доцент ОСГН ШБИП ТПУ | Маланина Вероника Анатольевна | к.э.н., доцент |  | 08.07.2022 |

**Задание принял к исполнению студент:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Группа** | **ФИО** | **Подпись** | **Дата** |
| О-5КМ01 | Чернобров Михаил Евгеньевич |  | 08.07.2022 |

**ЗАДАНИЕ ДЛЯ РАЗДЕЛА**

**«СОЦИАЛЬНАЯ ОТВЕТСТВЕННОСТЬ»**

Студенту:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Группа** | | **ФИО** | | |
| О-5КМ01 | | Черноброву Михаилу Евгеньевичу | | |
| **Школа** | **Инженерная школа энергетики** | | **Отделение (НОЦ)** | **Отделение электроэнергетики и электротехники** |
| **Уровень образования** | магистратура | | **Направление/ специальность** | 09.04.03 Прикладная информатика |

Тема ВКР:

|  |  |  |
| --- | --- | --- |
| **Разработка программного обеспечения для формирования эквивалентных математических моделей солнечных электростанций на основе данных информационной модели энергосистемы** | | |
| **Исходные данные к разделу «Социальная ответственность»:** | | |
| **Введение**   * Характеристика объекта исследования (вещество, материал, прибор, алгоритм, методика) и области его применения; * Описание рабочей зоны (рабочего места) при разработке проектного решения/при эксплуатации. | *Объект исследования:* программная реализация алгоритма эквивалентирования моделей солнечных электростанций.  *Область применения:* программные средства формирования расчетных моделей установившихся режимов электроэнергетических систем.  *Рабочая зона:* офис.  *Размеры помещения:* 52,8 кв. м.  *Количество и наименование оборудования рабочей зоны:* 1 шт. персональный компьютер; 1 шт. компьютерный стол; 1 шт. офисное кресло.  *Рабочие процессы, связанные с объектом исследования, осуществляющиеся в рабочей зоне:* проектирование и разработка программного обеспечения. | |
| Перечень вопросов, подлежащих исследованию, проектированию и разработке: | | |
| **1. Правовые и организационные вопросы обеспечения безопасности при разработке проектного решения:**   * Специальные (характерные при эксплуатации объекта исследования, проектируемой рабочей зоны) правовые нормы трудового законодательства; * Организационные мероприятия при компоновке рабочей зоны. | * Трудовой кодекс Российской Федерации от 30.12.2001 N 197-ФЗ (ред. от 29.12.2020); * ГОСТ 12.2.032-78 ССБТ Рабочее место при выполнении работ сидя; * ГОСТ Р 50923-96 Дисплеи. Рабочее место оператора. Общие эргономические требования и требования к производственной среде. Методы измерения. | |
| **2. Производственная безопасность при разработке проектного решения:**   * Анализ выявленных вредных и опасных производственных факторов; * Расчет уровня опасного или вредного производственного фактора. | Вредные факторы:   * Производственные факторы, связанные с аномальными микроклиматическими параметрами воздушной среды на местонахождении работающего; * Повышенный уровень шума; * Отсутствие или недостаток необходимого искусственного освещения; * Статические физические перегрузки, связанные с рабочей позой; * Нервно-психические перегрузки, связанные с напряженностью трудового процесса.   Опасные факторы:   * Производственные факторы, связанные с электрическим током, вызываемым разницей электрических потенциалов, под действие которого попадает работающий.   Средства коллективной и индивидуальной защиты:   * Системы кондиционирования; * Системы отопления; * Источники света; * Устройства защитного заземления.   Расчетный фактор:   * Отсутствие или недостаток необходимого искусственного освещения. | |
| **3. Экологическая безопасность при разработке проектного решения** | *Воздействие на литосферу:* твердые бытовые отходы.  *Воздействие на гидросферу:* бытовые сточные воды. | |
| **4. Безопасность в чрезвычайных ситуациях** **при разработке проектного решения** | *Возможные ЧС*: землетрясение, пожар.  *Наиболее типичная*: ЧС: пожар. | |
| **Дата выдачи задания для раздела по линейному графику** | | 08.07.2022 |

**Задание выдал консультант:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Должность** | **ФИО** | **Ученая степень, звание** | **Подпись** | **Дата** |
| доцент ООД ШБИП ТПУ | Антоневич Ольга Алексеевна | к.б.н., доцент |  | 08.07.2022 |

**Задание принял к исполнению студент:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Группа** | **ФИО** | **Подпись** | **Дата** |
| О-5КМ01 | Чернобров Михаил Евгеньевич |  | 08.07.2022 |



Школа: Инженерная школа энергетики

Направление подготовки (ООП/ОПОП): 09.04.03 Прикладная информатика

Уровень образования: магистратура

Отделение школы (НОЦ): Электроэнергетики и электротехники

Период выполнения: весенний/осенний семестр 2022/2023 учебного года

**КАЛЕНДАРНЫЙ РЕЙТИНГ-ПЛАН**

**выполнения выпускной квалификационной работы**

Обучающийся:

|  |  |
| --- | --- |
| **Группа** | **ФИО** |
| О-5КМ01 | Чернобров Михаил Евгеньевич |

Тема работы:

|  |
| --- |
| Разработка программного обеспечения для формирования эквивалентных математических моделей солнечных электростанций на основе данных информационной модели энергосистемы |

|  |  |
| --- | --- |
| Срок сдачи студентом выполненной работы: | 20.01.2023 |

|  |  |  |
| --- | --- | --- |
| **Дата контроля** | **Название раздела (модуля) /  вид работы (исследования)** | **Максимальный балл раздела (модуля)** |
| 16.09.2022 | Анализ предметной области | 10 |
| 21.10.2022 | Проектирование программного обеспечения | 15 |
| 23.12.2022 | Программная реализация | 25 |
| 30.12.2022 | Тестирование программного обеспечения | 10 |
| 13.01.2023 | Разработка руководства пользователя | 5 |
| 15.01.2023 | Выполнение задания по разделу «Социальная ответственность» | 10 |
| 18.01.2023 | Выполнение задания по разделу «Финансовый менеджмент» | 10 |
| 20.01.2022 | Оформление пояснительной записки выпускной квалификационной работы | 15 |

**СОСТАВИЛ:**

**Руководитель ВКР**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Должность** | **ФИО** | **Ученая степень, звание** | **Подпись** | **Дата** |
| доцент ОЭЭ ИШЭ ТПУ | Прохоров А.В. | к.т.н., доцент |  | 08.07.2022 |

**Консультант** (при наличии)**:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Должность** | **ФИО** | **Ученая степень, звание** | **Подпись** | **Дата** |
| доцент КСУП ТУСУР | Калентьев А.А. | к.т.н., доцент |  | 08.07.2022 |

**СОГЛАСОВАНО:**

**Руководитель ООП/ОПОП:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Должность** | **ФИО** | **Ученая степень, звание** | **Подпись** | **Дата** |
| доцент ОЭЭ ИШЭ ТПУ | Прохоров А.В. | к.т.н., доцент |  | 08.07.2022 |

**Обучающийся:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Группа** | **ФИО** | **Подпись** | **Дата** |
| О-5КМ01 | Чернобров Михаил Евгеньевич |  | 08.07.2022 |

# РЕФЕРАТ

Выпускная квалификационная работа на тему «Разработка программного обеспечения для формирования эквивалентных математических моделей солнечных электростанций на основе данных информационной модели энергосистемы» состоит из: XXX страниц, XX рисунков, XX таблиц, XX источника и X приложений.

Ключевые слова: программное обеспечение; информационная модель; расчетная модель; солнечная электростанция; коллекторная сеть; эквивалентирование.

Объект исследования – проектирование и разработка прикладного программного обеспечения в сфере электроэнергетики;

Предмет исследования – деловой процесс и программные средства формирования и актуализации расчетных и информационных моделей электроэнергетических систем.

Цель работы – автоматизация процедуры формирования эквивалентного представления солнечных электростанций в составе расчетной модели установившихся режимов на основе данных информационной модели электроэнергетической системы.

Основным специальным методом исследования выступает формализация. В ходе формализации осуществляется алгоритмизация и разработка программного средства посредством искусственного формализованного языка (языка программирования).

В качестве инструментов исследования выступают: интегрированная среда разработки Visual Studio, язык программирования C# и его платформа .NET, система контроля версий git и программный комплекс RastrWin3.

Область применения – формирование и актуализация расчетных моделей установившихся режимов.

Практическая значимость работы выражена в:

* снижении трудозатрат и минимизации ошибок за счет автоматизации процедуры эквивалентирования;
* обеспечении актуальности параметров расчетной модели за счет использования информационной модели ОИК в качества источника данных;
* и как результат, в снижении операционных издержек организации на поддержание расчетной модели в актуальном состоянии.

Новизна работы обеспечивается реализацией алгоритма эквивалентирования из условия суммарных потерь в элементах коллекторной сети солнечной электростанции посредством реализации алгоритма поиска пути в графе методом Дейкстры.

Разработанное программное обеспечение может быть внедрено в ИТ‑инфраструктуру ОДУ Сибири и РДУ операционной зоны ОДУ, содержащих в составе энергосистем солнечные электростанции, являющихся объектами диспетчеризации.

# ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

**ОГЛАВЛЕНИЕ**

[РЕФЕРАТ 10](#_Toc124765876)

[ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ 12](#_Toc124765877)

[ВВЕДЕНИЕ 15](#_Toc124765878)

[1. Анализ предметной области 18](#_Toc124765879)

[1.1 Формирование расчетной модели установившихся режимов 19](#_Toc124765880)

[1.2 Информационная модель солнечных электростанций 22](#_Toc124765881)

[1.2.1 Объект «электростанция» 22](#_Toc124765882)

[1.2.2 Объект «генератор» (инвертор – фотоэлектрические модули) 24](#_Toc124765883)

[1.2.3 Объект «фидер коллекторной сети» 29](#_Toc124765884)

[1.2.4 Объект «преобразовательный трансформатор» (трансформатор инверторной станции) 31](#_Toc124765885)

[1.3 Выводы по разделу 33](#_Toc124765886)

[2. Проектирование программного обеспечения 34](#_Toc124765887)

[2.1 Варианты интеграционного взаимодействия 34](#_Toc124765888)

[2.1.1 Эквивалентирование расчетной модели 34](#_Toc124765889)

[2.1.2 Предварительное эквивалентирование информационной модели 35](#_Toc124765890)

[2.1.3 Эквивалентирование расчетной модели по данным информационной модели 37](#_Toc124765891)

[2.2 Алгоритм эквивалентирования 38](#_Toc124765892)

[2.3 Определение требований к программному обеспечению 39](#_Toc124765893)

[2.4 Варианты использования 40](#_Toc124765894)

[2.5 Диаграмма компонентов 43](#_Toc124765895)

[2.6 Макет пользовательского интерфейса 45](#_Toc124765896)

[2.7 Выводы по разделу 49](#_Toc124765897)

[3. Программная реализация 51](#_Toc124765898)

[3.1 Выбор инструментов 51](#_Toc124765899)

[3.1 Model (модель данных) 52](#_Toc124765900)

[3.3 RastrSupplier 56](#_Toc124765901)

[3.4 ViewModel (модель представления) 58](#_Toc124765902)

[3.5 Equivalentator 60](#_Toc124765903)

[3.6 View (представление) 61](#_Toc124765904)

[3.7 Выводы по разделу 67](#_Toc124765905)

[4. Тестирование 68](#_Toc124765906)

[СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ 69](#_Toc124765907)

[Приложение А 71](#_Toc124765908)

[Приложение Б 74](#_Toc124765909)

[Приложение В 75](#_Toc124765910)

[Приложение Г 88](#_Toc124765943)

# ВВЕДЕНИЕ

Согласно прогнозу, представленному в Генеральной схеме размещения объектов Единой энергетической системы (ЕЭС) до 2035 года, в базовом варианте предполагается ввод до 1,8 млн. кВт генерирующего оборудования, функционирующего на основе возобновляемых источников энергии [1].

Основные задачи при планировании и оперативном управлении электроэнергетическим режимом решаются на основе предварительных расчетов установившихся режимов (УР) с использованием моделей энергосистем (ЭС).

Для достоверного представления регулировочного диапазона солнечных электростанций (СЭС) по реактивной мощности, они должны быть представлены в расчетных и информационных моделях (ИМ) электроэнергетических систем с учетом параметров элементов коллекторной сети. Однако в данный момент это не обеспечивается в рамках деловых процессов в сфере оперативно‑диспетчерского управления.

При этом в виду сложности топологии коллекторной сети, в расчетных моделях станции рекомендуется представлять эквивалентной структурой [2].

Рост числа СЭС и необходимость их эквивалентного представления в моделях электроэнергетических систем увеличивают трудозатраты филиалов АО «СО ЕЭС» в рамках деловых процессов формирования и актуализации расчетных моделей (РМ) установившихся режимов.

Таким образом, существует потребность в автоматизации указанных деловых процессов, что является целью данной работы.

Более того среди ключевых направлений программы инновационного развития Системного оператора (АО «СО ЕЭС») до 2025 года определены следующие приоритетные направления [3]:

* разработка новых или существенное усовершенствование действующих моделей, методов и способов управления и планирования электроэнергетическими режимами ЕЭС России. Внедрение инновационных решений в практику оперативно-диспетчерского управления в электроэнергетике;
* внедрения в производство новых инновационных решений в информационных технологиях (ИТ), современного высокопроизводительного ИТ‑оборудования и средств автоматизации производственных и деловых процессов.

Сложившаяся ИТ-инфраструктура АО «СО ЕЭС» в части актуализации и формирования моделей предоставлена:

* информационной моделью оперативного-информационного комплекса (ОИК) СК-11:
* основанной на международном стандарте Common Information Model (CIM) в формате XML;
* актуализируемой посредством CIM-портала;
* расчётной моделью установившихся режимов программного комплекса «RastrWin3» (ПК «RastrWin3»), экспортируемой посредством программного обеспечения «Интеграции» (ПО «Интеграции») с возможностью наложения:
* режимных модификаций по дням контрольного замера (ДКЗ);
* ремонтных заявок из программного комплекса «Заявки, Ремонты, Перечни» (CIM ЗРП).

В таком случае, в виду приведенных технических возможностей, становится актуальной реализация программного обеспечения для получения в автоматизированном режиме РМ установившихся режимов, содержащих эквивалентные модели СЭС.

Для достижения поставленной цели были решены следующие задачи:

1. Проанализированы:

* Процедура формирования РМ установившихся режимов электроэнергетических систем;
* Представление СЭС в ИМ электроэнергетической системы.

1. Разработаны:

* Варианты интеграционного взаимодействия;
* Алгоритм эквивалентирования;

1. Определены требования к разрабатываемому ПО;
2. Осуществлено концептуальное проектирование;
3. Выполнена программная реализация:

* Модели данных;
* Методов взаимодействия с ПК «RastrWin3»;
* Модели представления;
* Расчетного модуля (модуля эквивалентирования);
* Графического интерфейса.

1. Осуществлено автономное тестирование ПО;
2. Оформлена техническая документация.

# 1. Анализ предметной области

Общая информационная модель (англ. CIM – Common Information Model) – абстрактная цифровая модель системы, описывающая ее основные элементы, их свойства и связи между ними в виде общепризнанных и одинаково понимаемых определений и понятий.

Использование CIM в электроэнергетике предоставляет возможность эффективной интеграции разнородных автоматизированных систем и обеспечивает унифицированный обмен информацией.

Международный стандарт IEC 61970 [4] представляет модель, описывающую оборудование и другие элементы энергосистемы в виде объектов классов, их свойств (атрибутов) и связей (ассоциаций).

В данный момент в АО «СО ЕЭС» осуществлены этапы создания единой информационной модели (ЕИМ) ЕЭС России, внедрения процессов поддержания ЕИМ в актуальном состоянии, введения в промышленную эксплуатации оперативно-информационного комплекса нового поколения (ОИК НП) СК-11, что в свою очередь позволило унифицировать информационный обмен между субъектами и диспетчерскими центрами (ДЦ), а также определить ОИК как единый источник данных для ряда. смежных систем.

В активном состоянии находится интеграция информационно-управляющих систем (ИУС) и программных комплексов (ПК) через автоматизированную интеграционную платформу (АИП), выступающую в качестве сервисной шины.

## 1.1 Формирование расчетной модели установившихся режимов

Согласно Регламенту [5] расчеты установившихся режимов и статической устойчивости осуществляются на единых расчетных моделях (ЕРМ) с использованием ПК «RastrWin3».

При этом согласно Регламенту [6] в состав перечня работ, выполняемых в процессе регламентной актуализации ИМ входит формирование РМ с использованием ПО «Интеграция» в ПК «RastrWin3».

Таким образом, источником данных для РМ является ИМ электроэнергетической системы.

Для актуализации ИМ субъекты электроэнергетики посредством веб‑интерфейса СIM-портала передают XML‑макеты с актуальной информацией о параметрах и характеристиках оборудования.

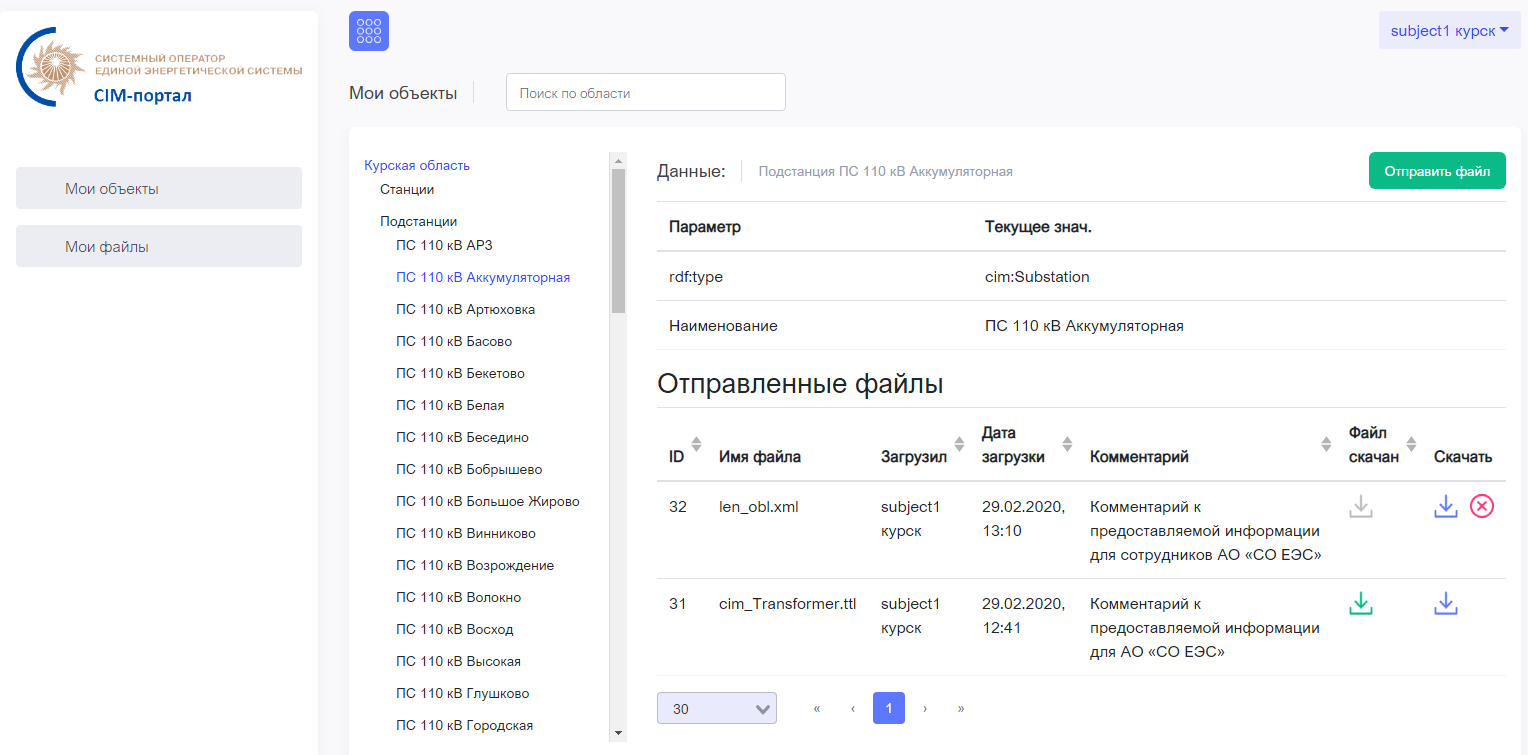


Рисунок 1.1 – Веб-интерфейс CIM-портала []

Экспорт РМ, как было упомянуто ранее, осуществляется с применением ПО «Интеграции», которая является компонентом ПК «RastrWin3».

В ходе экспорта расчетной модели ПО «Интеграции» позволяет:

* Выбрать определенную модель и версию ИМ;
* Загрузить режимные модификации, например по дням ДКЗ;
* Загрузить диспетчерские и ремонтных заявки из CIM-ЗРП.

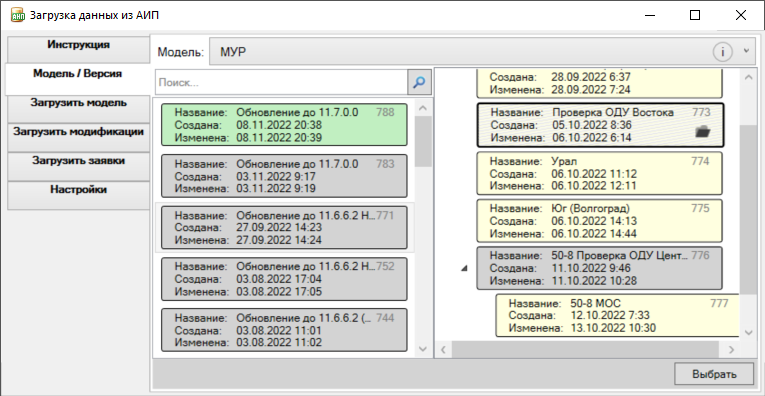


Рисунок 1.2 – Интерфейс ПО «Интеграции»

RastrWin3 позволяет работать с моделями ЭС в топологическом («узлы‑ветви») и коммутационном («шины-выключатели») представлениях.

Работа с моделями в топологическом представлении осуществляется посредством расчетного блока (ASTRA).

Для работы с моделями в коммутационном представлении используются сервер коммутационного слоя (SUB). Благодаря этому обеспечивается работа с моделями в формате CIM-XML [7].

Часть функций SUB в контексте взаимодействия с АИП требуют расширения, поэтому при установке ПО «Интеграции», последняя конфигурирует коммутационный слой, исключая последующую работу с моделями в формате CIM‑XML в виде локальных файлов.

Взаимодействие с АИП основано на клиент-сервисных запросах по протоколу SOAP (англ. Simple Object Access Protocol). Для этого на стороне АИП развернуты сервисы: управления контекстом (УК), доступа к информационной модели (ДИМ), библиотеки электрических режимов (БЭР) и формализованных заявок (ФЗ).

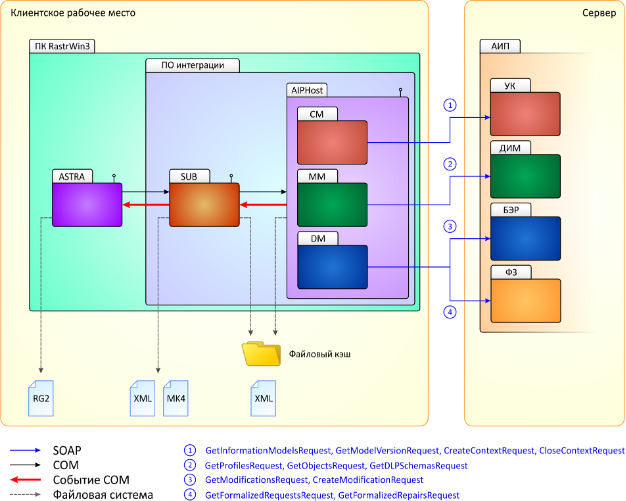


Рисунок 1.3 – Компоненты ПО «Интеграции» [7]

## 1.2 Информационная модель солнечных электростанций

Адаптация международного стандарта общей информационной модели электроэнергетических систем IEC 61970 [] представлена серией государственных отраслевых стандартов:

* ГОСТ Р 58651 Информационная модель «Основные положения» [8];
* ГОСТ Р 58651.2-2019 «Базисный профиль информационной модели» [9];
* ГОСТ Р 58651.3-2020 «Профиль информационной модели линий электропередачи и электросетевого оборудования напряжением 110‑750 кВ» [10];
* ГОСТ Р 58651.4-2020 «Профиль информационной модели генерирующего оборудования» [11];
* ГОСТ Р 58651.6-2022 «Профиль информационной модели линий электропередачи и электросетевого оборудования напряжением 0,4‑35 кВ» [12].

### 1.2.1 Объект «электростанция»

ГОСТ Р 58651.2-2019 и его расширение ГОСТ Р 58651.4-2020 определяют порядок моделирования объектов, выступающих контейнерами для оборудования электростанций, посредством экземпляров классов:

* «Production::HydroPowerPlant» (Гидроэлектростанция);
* «Production::CogenerationPlant» (Группа генерирующих установок, предназначенных для выработки электроэнергии и технологического пара);
* «Production::CombinedCyclePlant» (Группа генерирующих установок с комбинированным циклом).

При этом отраслевые стандарты не определяют классы для представления контейнеров станций ВИЭ, в частности СЭС.

Более того актуальная каноническая модели (КМ) АО «СО ЕЭС» также не расширяет базовый профиль ИМ в части представления СЭС. Сейчас данные станции представлены как объекты «Production::CogenerationPlant».

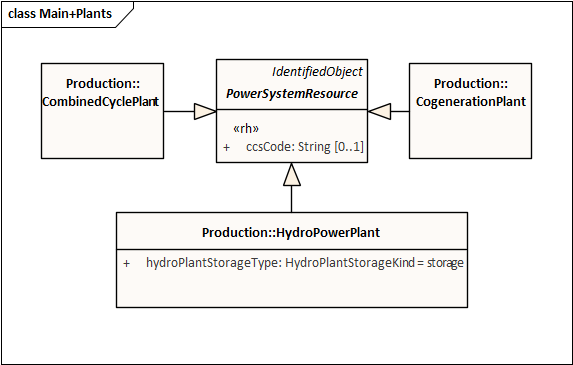


Рисунок 1.4 – Диаграмма классов контейнеров электростанций согласно отраслевым стандартам и актуальной канонической модели



Рисунок 1.5 – Фрагмент XML-структуры объекта станции  
на примере Ининской СЭС

Важно отметить, что актуальная версия международного стандарта IEC 61970 (CIM 17 v38) в своем расширении профиля определяет объекты «ExtEuProduction::SolarPowerPlant» для моделирования контейнеров СЭС [4]. При этом в данном классе не определены уникальные атрибуты, относительно наследуемых классов.

Таким образом, моделирование СЭС объектами «Production::CogenerationPlant» является приемлемым. Однако имеет место нарушение семантики.

### 1.2.2 Объект «генератор» (инвертор – фотоэлектрические модули)

ГОСТ Р 58651.2-2019 и его расширение ГОСТ Р 58651.4-2020 устанавливают подход в представлении генерирующего оборудования электростанций.

Так фотоэлектрические модули (ФЭМ) должны быть представлены объектами класса «Production::PhotoVoltaicUnit» (Фотоэлектрический солнечный модуль или группа модулей) – наследник абстрактного класса «Production::PowerElectronicsUnit» (Генерирующая единица, накопитель или устройство, подключенные к сети переменного тока с использованием силовой электроники).

Инвертор должен быть представлен объектами класса «Wires::PowerElectronicsConnection» (Устройство на базе силовой электроники для включения в сеть солнечных модулей, ветряных энергоустановок) – наследник абстрактного класса «Wires::RegulatingCondEq» (Регулирующее электропроводящее оборудование).

При этом «Wires::PowerElectronicsConnection» агрегирует «Production::PhotoVoltaicUnit».

Диаграмма классов генерирующего оборудования, соответствующая отраслевым стандартам приведена на рисунке 1.6.

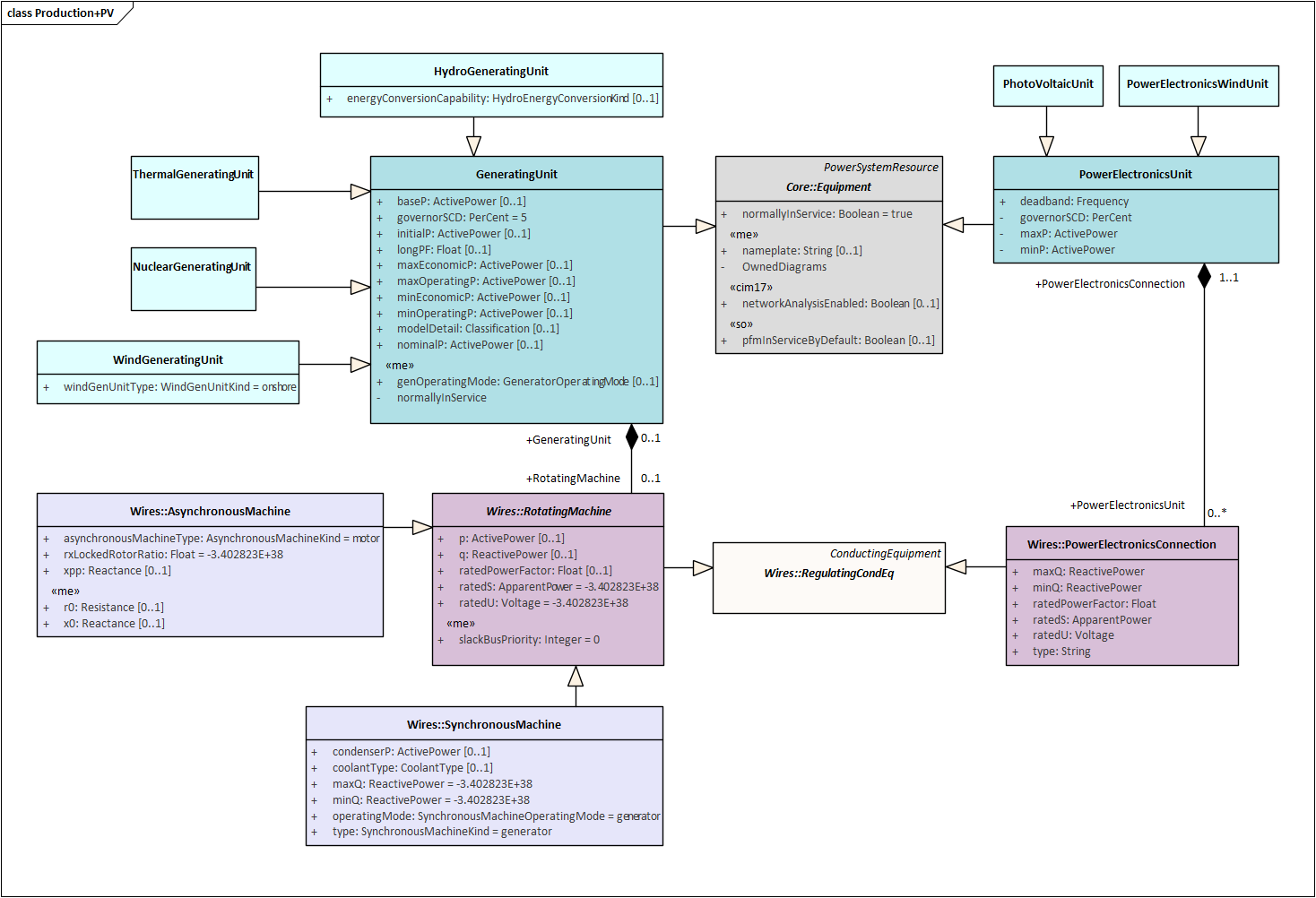


Рисунок 1.6 – Диаграмма классов генерирующего оборудования согласно отраслевым стандартам

Однако актуальная каноническая модель АО «СО ЕЭС» не располагает объектами для моделирования генерирующего оборудования СЭС.

Фотоэлектрические модули моделируются объектами класса «Production::ThermalGeneratingUnit» (Тепловой энергоблок) – наследник абстрактного класса «Production::GeneratingUnit» (Вращающиеся электрические машины).

А инвертор – объектами класса «Wires::SynchronousMachine» (Синхронная машина), являющегося наследником абстрактного класса «Wires::RotatingMachine» (Вращающаяся машина).

«Wires::RotatingMachine» так же агрегирует «Production::ThermalGeneratingUnit».

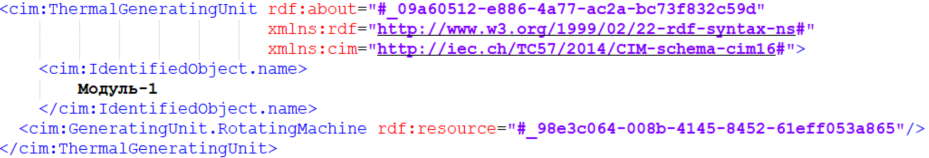


Рисунок 1.7 – Фрагмент XML-структуры генератора  
(инвертор – фотоэлектрические модули) на примере Ининской СЭС

Диаграмма классов генерирующего оборудования, соответствующая актуальной канонической модели АО «СО ЕЭС» приведена на рисунке 1.8.

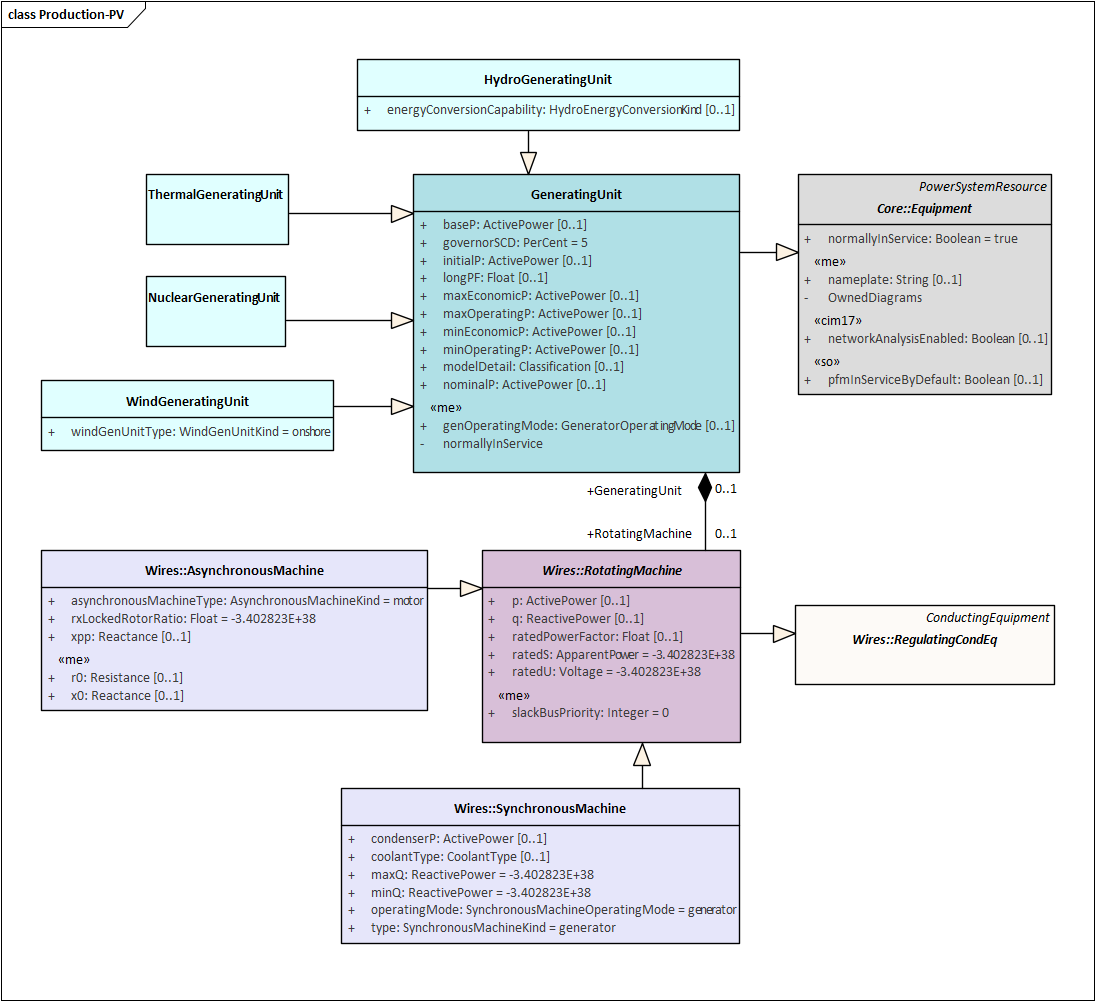


Рисунок 1.8 – Диаграмма классов генерирующего оборудования  
актуальной канонической модели

При формальном сопоставлении атрибутов между парами классов «Production::ThermalGeneratingUnit» ‑ «Production::PhotoVoltaicUnit» и «Wires::RotatingMachine» ‑ «Production::PowerElectronicsConnection» (Приложение А) можно сделать заключение, что текущий подход в моделировании генерирующего оборудования СЭС является приемлемым.

Также важно отметить, что при приведении моделирования в соответствие к отраслевому стандарту, потребуется расширение профиля КМ, так как для ассоциации объектов «Production::PhotoVoltaicUnit» отсутствуют объекты станций.

Ассоциация объектов «Production::PhotoVoltaicUnit» с объектами станций приведена на рисунке 1.9.

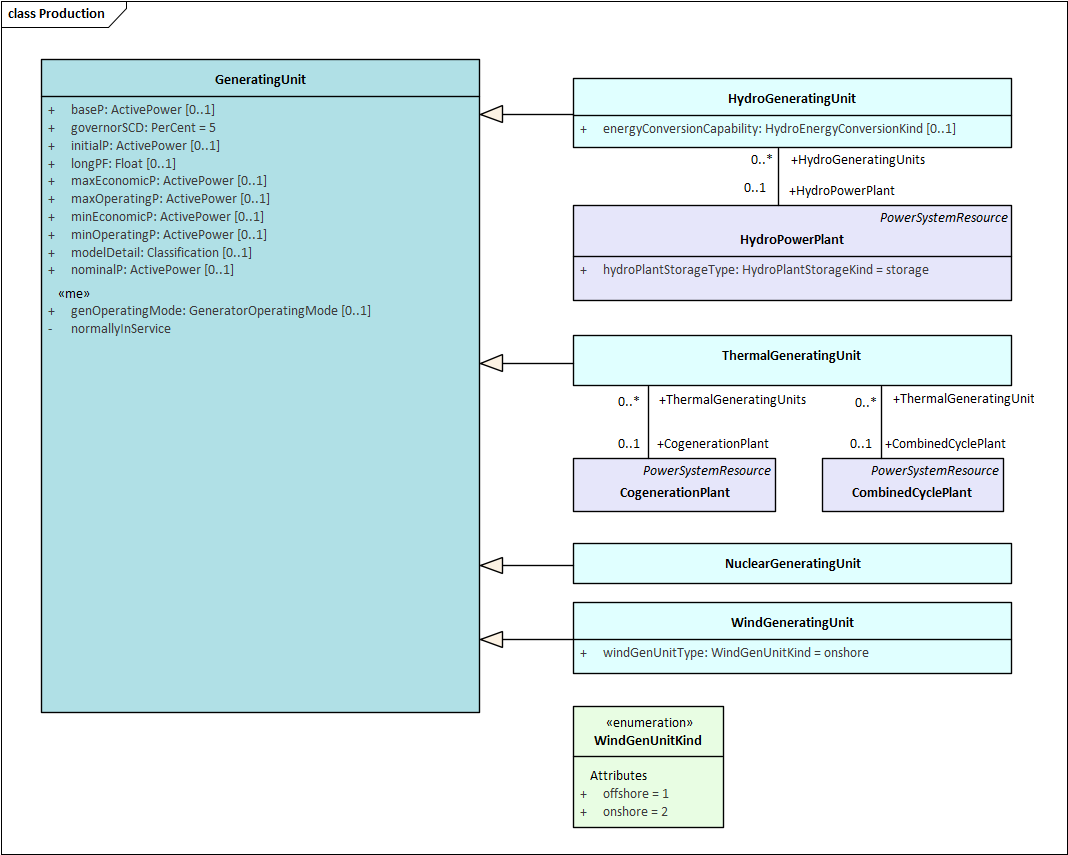


Рисунок 1.9 – Диаграмма классов ассоциации объектов генерирующего оборудования и объектов-контейнеров электростанций

### 1.2.3 Объект «фидер коллекторной сети»

ГОСТ Р 58651.6-2022 устанавливает подход представления фидеров, объектами класса «Wires::Feeder» (Питающая линия — контейнер оборудования, соединяющего распределительное устройство (РП) питающей подстанции с одним или несколькими РП устройствами, либо потребителями).

При таком подход объекты-контейнеры ПС могут ассоциировать объекты-контейнеры фидеров, при этом одни и те же контейнеры фидеров могут быть ассоциированы с несколькими объектами ПС.

Диаграмма классов проводящего оборудования (фидеры, линии и т.п.), соответствующая отраслевым стандартам приведена на рисунке 1.10.

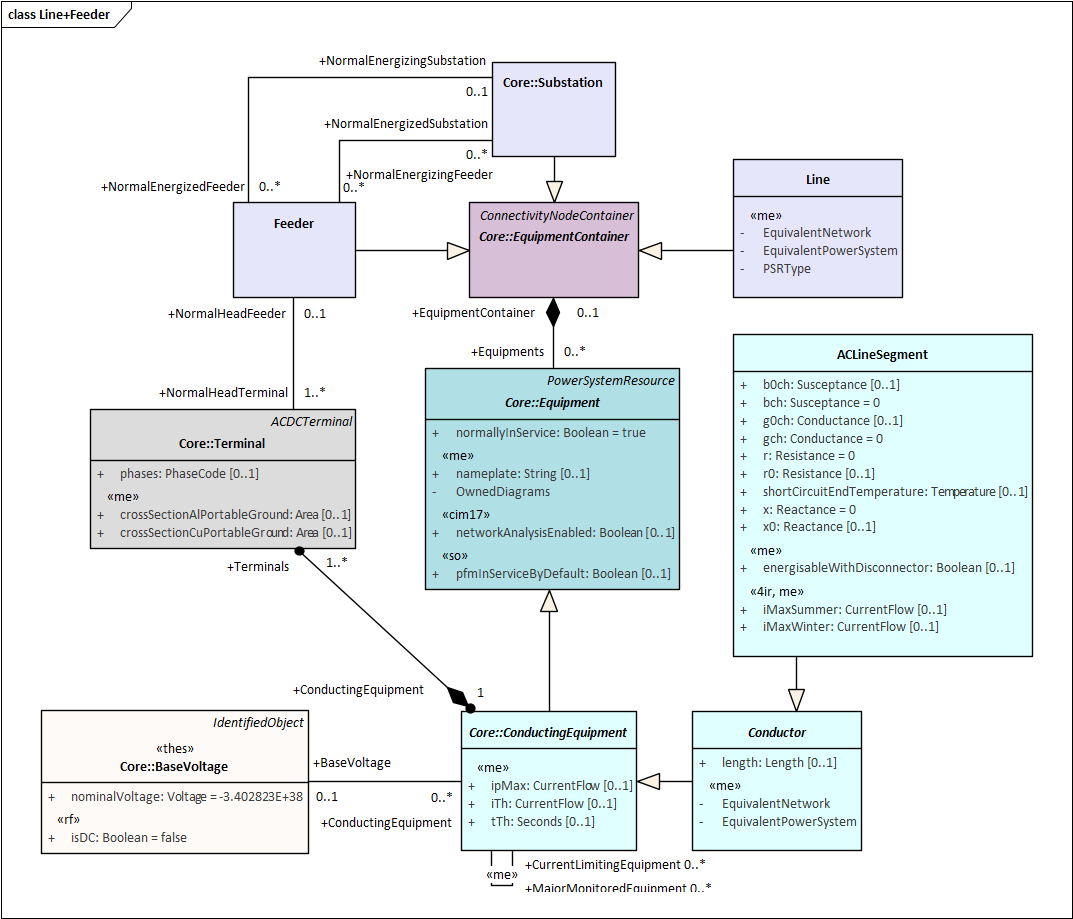


Рисунок 1.10 – Диаграмма классов проводящего оборудования  
(фидеры, линии и т.п.) согласно отраслевым стандартам

Объекты «Wires::Feeder», по аналогии с «Wires::Line» (ЛЭП), композируют объекты «ACLineSegment» (Участок (сегмент) линии переменного тока) – множественного наследниках класса «Core::Equipment» (Оборудование). При этом «Wires::Feeder» не определяют уникальные атрибуты, относительно «Wires::Line».

В актуальной версии канонической модели АО «СО ЕЭС» отсутствует расширение профиля для моделирования объектов фидеров коллекторной сети.

Диаграмма классов проводящего оборудования (фидеры, линии и т.п.), соответствующая актуальной версии канонической модели приведена на рисунке 1.12.

При это и сами модели солнечных электростанций в ИМ не содержат элементов коллекторной сети, в частности фидеров среднего напряжения, необходимость наличия которых ранее упоминалась.

Однако в ИМ все таки имеется практика моделирования фидеров среднего напряжения объектами-контейнерами «Wires::Line».

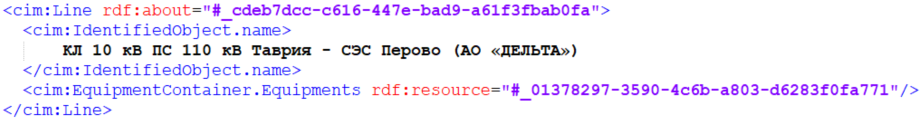


Рисунок 1.11 – Фрагмент XML-структуры фидера   
на примере КЛ 10 кВ ПС 110 кВ Таврия – СЭС Перово

В таком случае использовании контейнера «Wires::Line» для моделирования фидеров коллекторной сети СЭС – приемлемо, однако фидеры будут обособлены от конкретной станции (подстанции).

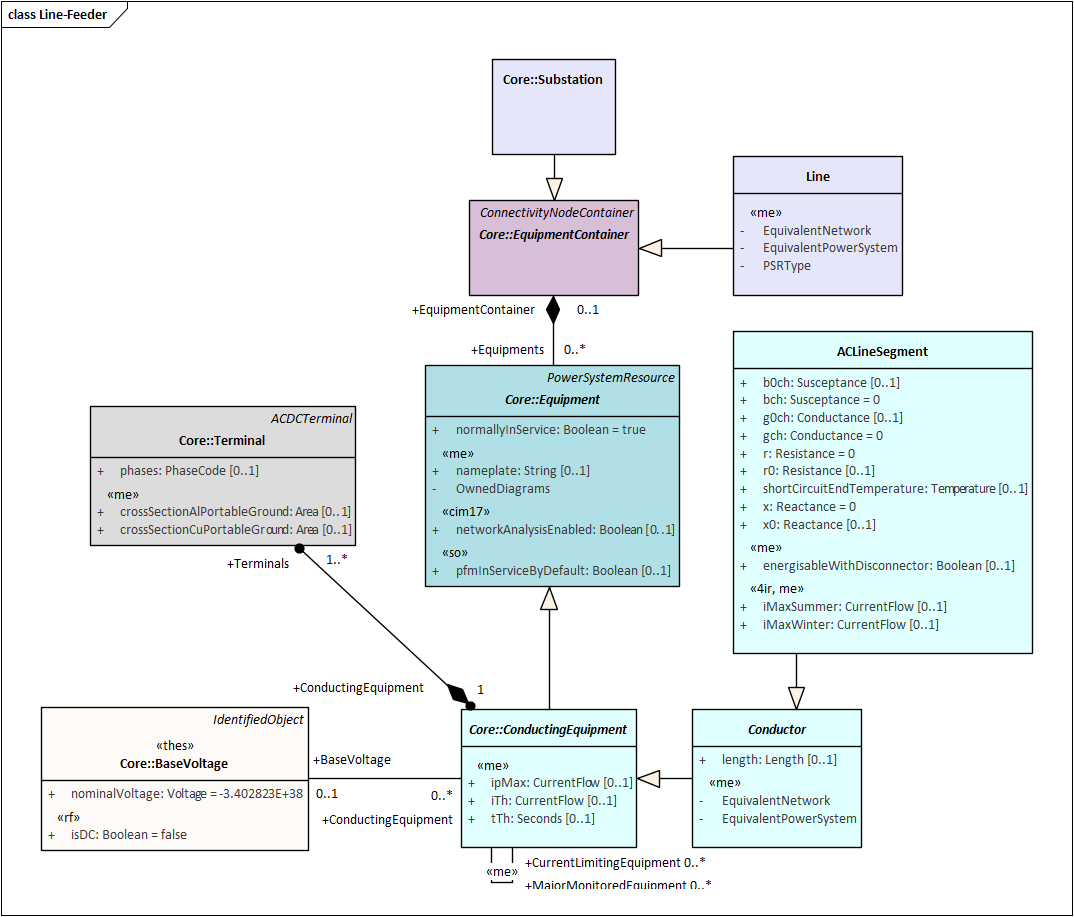


Рисунок 1.12 – Диаграмма классов проводящего оборудования

(фидеры, линии и т.п.) согласно актуальной версии канонической модели

### 1.2.4 Объект «преобразовательный трансформатор» (трансформатор инверторной станции)

ГОСТ Р 58651.2-2019 определяет порядок моделирования силовых трансформаторов объектами класса «Wire::PowerTransformer» (Силовой трансформатор), выступающими контейнерами для электрических выводов трансформаторов (ТР) «Wire::PowerTransformerEnd» (Электрический вывод силового трансформатора).

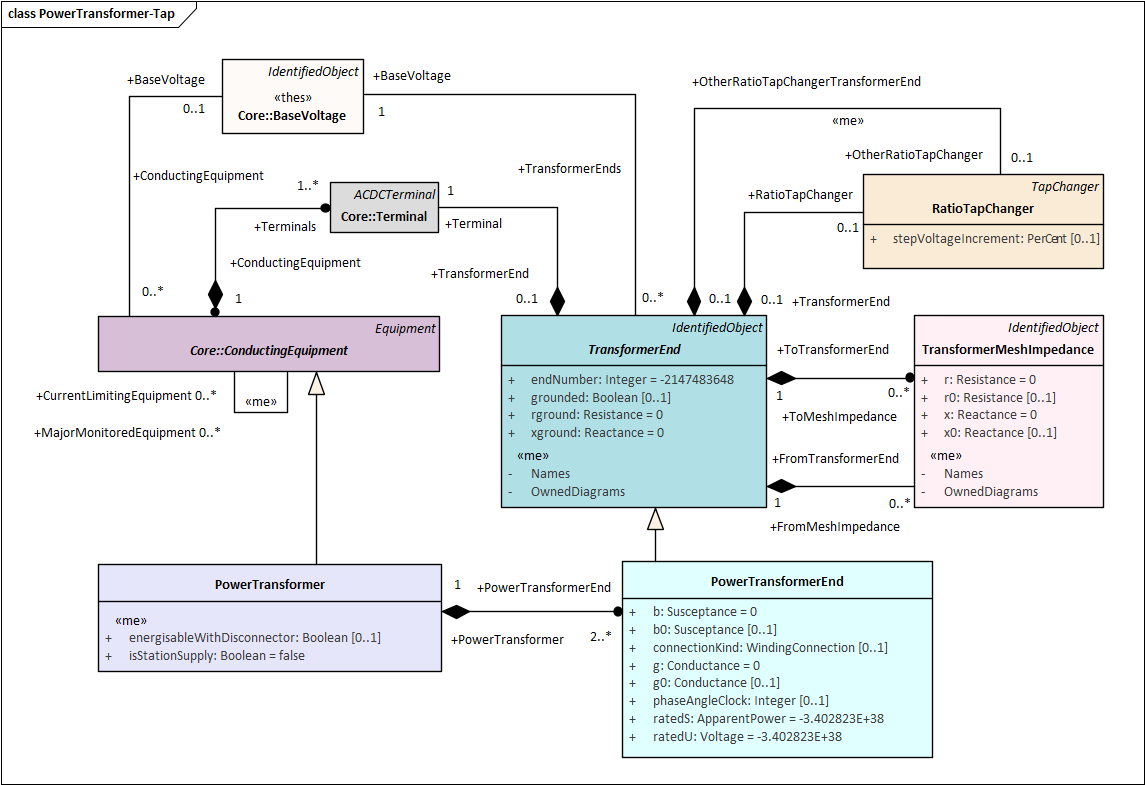


Рисунок 1.13 – Диаграмма классов проводящего оборудования (силовые трансформаторы) согласно отраслевым стандартам  
и актуальной версии канонической модели

Электрические выводы в свою очередь композируют устройства регулирования «Wires::RatioTapChanger» (Регулятор коэффициента трансформации) и характеристики обмоток «Wires::TrasformerMeshImpedance» (Сопротивления продольной ветви).

Модели СЭС в ИМ не содержат преобразовательных трансформаторов, необходимость наличия которых ранее упоминалась.

Однако представление их в ИМ не требует особенного подхода. Наиболее интересующие их характеристики (сопротивление и коэффициент трансформации) могут быть представлены посредством обычных силовых трансформаторов.

Также актуальная версия международного стандарта IEC 61970 (CIM 17 v38) не определяет отдельных классов для их моделирования [4].

## 1.3 Выводы по разделу

В результате анализа делового процесса формирования РМ установившихся режимов была определена ИТ-инфраструктура АО «СО ЕЭС», обеспечивающая его выполнение.

Определены имеющиеся ограничения, которые в последствие будут является основой для проработки вариантов интеграции разрабатываемого программного средства.

Так, например, ПО «Интеграции» используемое для экспорта РМ из ИМ, конфигурирует коммутационный слой ПК «RastrWin3», в результате чего исключена дальнейшая работа с моделями в формате CIM‑XML в виде локальных файлов.

В части анализа КМ и ее реализацией – ИМ, выступающей источником данных для РМ, определены технические возможности уточнения коллекторной сети СЭС.

Результат анализа и заключение с предложением о подходе моделирования объектов приведены в результирующей таблице 1.4.

Таблица 1.4 – Требования и фактический подход в моделировании элементов солнечных электростанций

|  |  |  |  |
| --- | --- | --- | --- |
| Объект | Требование ГОСТ | Фактическое представление в КМ | Предложение по моделированию в ИМ |
| СЭС | **×** | CogenerationPlant | CogenerationPlant |
| Генератор (фотоэлектрические модули - инвертор) | PhotoVoltaicUnit, PowerElectronics Connection | ThermalGenerating Unit,  SynchronousMachine, | ThermalGeneratingUnit,  SynchronousMachine, |
| Фидеры | Feeder | **×** | Line |
| Преобразовательные трансформаторы | **×** | **×** | PowerTransformer,  PowerTransformerEnd |

# 2. Проектирование программного обеспечения

UML (Unified Modeling Language) – унифицированный язык моделирования. Нотация UML представляет набор соглашений, предназначенных для систематизации процесса моделирования, и предоставления информации в ходе проектирования. Такой подход в значительной степени облегчает интерпретацию информации, документирование и взаимодействие между специалистами по проекту.

## 2.1 Варианты интеграционного взаимодействия

Исходя из ранее рассмотренного процесса формирования РМ и определенных технических возможностей для получения моделей с эквивалентными структурами СЭС из ИМ были рассмотрены три варианта интеграции в существующий деловой процесс.

### 2.1.1 Эквивалентирование расчетной модели

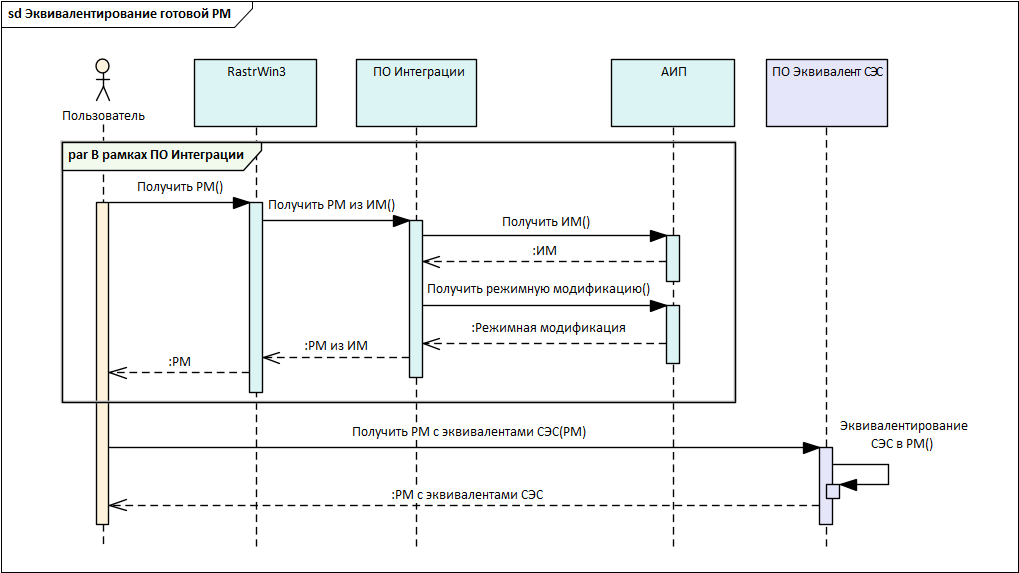


Рисунок 2.1 – Диаграмма последовательности.  
«Эквивалентирование расчетной модели»

При таком подходе разрабатываемое ПО будет осуществлять эквивалентирование моделей СЭС непосредственно в расчетной модели ПК «RastrWin3», предварительно полученной с применением ПО «Интеграции». Формат входного и выходного файла – «\*.rg2».

Среди преимуществ данного варианта:

* независимость от подхода в представлении СЭС в КМ и версии ПО «Интеграции», что исключает необходимость в изменении программного кода, в случае перехода на моделирования СЭС в ИМ согласно отраслевому стандарту;
* представления минимально жизнеспособного продукта (MVP – minimal value product) в ограниченные сроки.

Среди недостатков могут быть отмечены:

* работа с моделью энергосистемы в отличном от CIM‑XML формате, что не cответствует современным тенденциям развития автоматизированных систем диспетчерского управления (АСДУ), направленных на унификацию информационного обмена;
* наличие дополнительного ПО в ИТ-инфраструктуре организации, что увеличивает операционные издержки на сопровождение и поддержание его в актуальном и рабочем состоянии.

### 2.1.2 Предварительное эквивалентирование информационной модели

Согласно ранее приведенной информации, ПО «Интеграции» конфигурирует коммутационный слой ПК «RastrWin3», в дальнейшем выступая посредником между ИМ энергосистемы в формате CIM-XML и ее отображением в топологическом представлении.

Приведенный вариант подразумевает модификацию ПО «Интеграции», в результате которой последняя осуществляет эквивалентирование СЭС в ИМ до ее передачи на коммутационный слой RastrWin3.

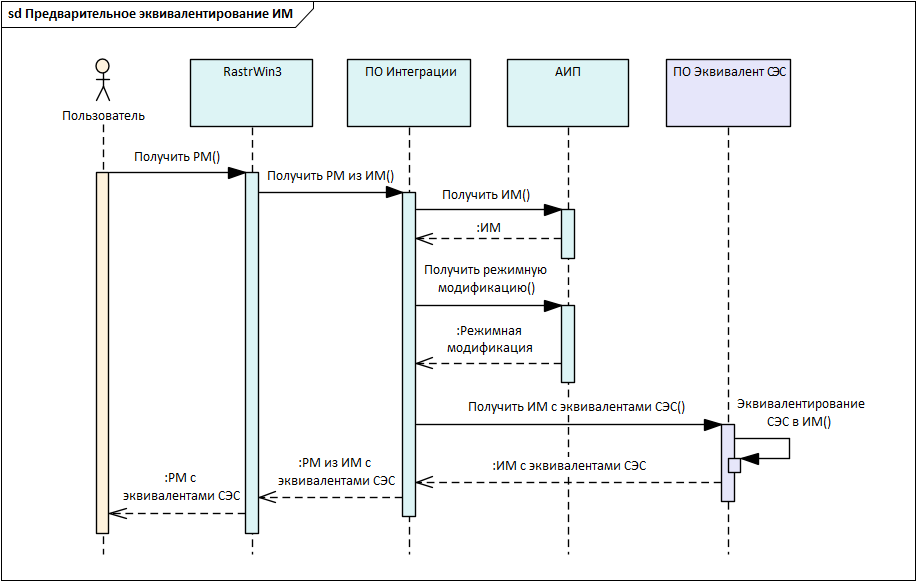


Рисунок 2.2 – Диаграмма последовательности  
«Эквивалентирование информационной модели»

Среди преимуществ эквивалентирования ИМ, относительно эквивалентирования РМ:

* работа с моделью энергосистемы в формате CIM XML, что наоборот соответствует тенденциям развития АСДУ;
* цельная архитектура ПО, что предоставляет относительно меньше накладные расходы на ее сопровождение.

Однако отсутствие программного интерфейса (API – application program interface) ПО «Интеграции», что является следствием сильной связности ее компонентов с ПК «RastrWin3», препятствует самостоятельной разработки программного средства в виде дополнительного модуля.

Доработка ПО «Интеграции» с целью реализации функционала эквивалентирования может быть рассмотрена в рамках модификации системы.

### 2.1.3 Эквивалентирование расчетной модели по данным информационной модели

Эквивалентирование РМ по данным ИМ является расширением варианта обычного эквивалентирования расчетной модели.

Предложения таким способом интеграция подразумевает загрузку в разрабатываемое ПО не всей модели ЭС (так же в формате «\*.rg2»), а только тех объектов, которые отнесены к СЭС.

Благодаря этому может быть повывшего удобство работы с ПО (выбора объектов подлежащих эквивалентированию), а также скорость загрузки данных в систему.

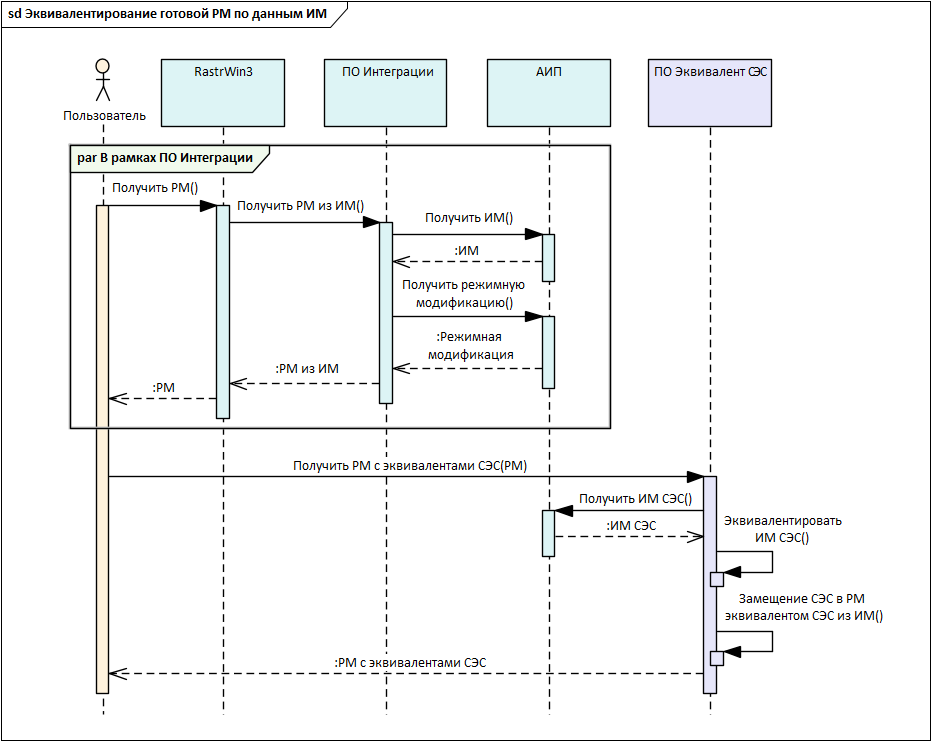


Рисунок 2.3 – Диаграмма последовательности  
«Эквивалентирование расчетной модели  
по данным информационной модели»

## 2.2 Алгоритм эквивалентирования

Метод эквивалентирования предлагаемый к реализации основан на расчете эквивалентного сопротивления из условия суммарных потерь в элементах коллекторной сети СЭС.

При этом, предварительно, необходимо восстановить граф сети, основой которого являются элементы электроэнергетической системы, содержащиеся в расчетной модели ПК «RastrWin3».

В качестве ребер (E) графа выступают ветви (линии), а в качестве вершин (V) – узлы (шины). Полученный граф можно охарактеризовать, как взвешенный и ненаправленный.

Представление энергосистемы в виде матрицы смежности (вариант представления графа) позволяет реализовать алгоритм поиска пути между парой вершин, который необходим для автоматизации алгоритма эквивалентирования.

Для поиска пути применяется алгоритм Дейкстры – поиск кратчайшего пути между двумя вершинами, реализуемый при неотрицательных весах ребер.

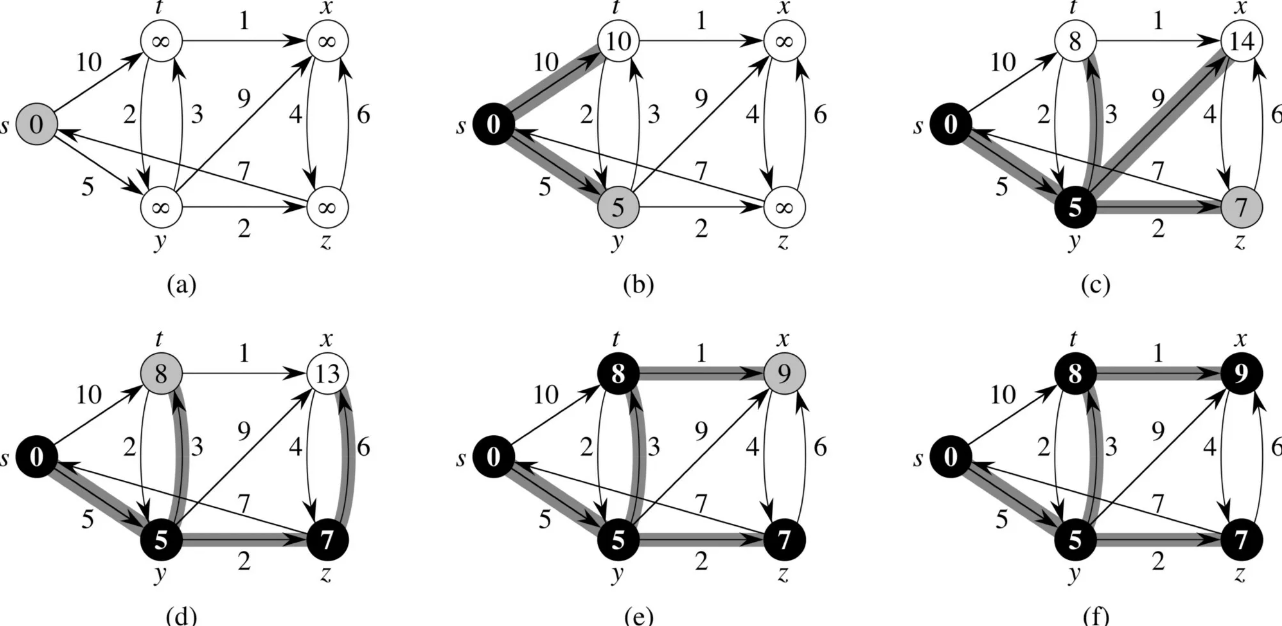


Рисунок 2.4 – Итерации (а – f) реализации алгоритма Дейкстры

Алгоритм присваивает вершинам V временные метки D(Vi). Метка вершины дает верхнюю границу длины пути от S (начальной вершины) к *i* вершине. Величины меток постепенно уменьшаются, и на каждом шаге итерации одна из временных меток становится постоянной. Это означает, что метка дает точную длину кратчайшего пути от *S* к рассматриваемой вершине.

Укрупненная диаграмма деятельности алгоритма эквивалентирования представлена в Приложении Б.

## 2.3 Определение требований к программному обеспечению

Требование – это объективно проверяемое самодостаточное утверждение, которое описывает некую потребность и не накладывает преждевременных технических ограничений.

Управление требованиями – процесс, включающий инженерию, проработку и приоритезацию требований, а также обеспечение соглашения об их изменении или условий достижения.

Техническое задание (ТЗ) – один из основных документов проектной документации, в котором фиксируются:

* цели, назначение и область применения разработки;
* представители исполнителя и заказчика, а также порядок их взаимодействия;
* технические требования и технико-экономические показатели;
* стадии и этапы разработки;
* порядок приема выполненных работ.

Для ТЗ определены отраслевые стандарты – ГОСТ 19.201 78 «Единая система программной документации. Техническое задание. Требования к содержанию и оформлению» [13] и ГОСТ 34.602-2020 «Техническое задание на создание автоматизированной системы» [14].

При этом приведенные стандарты не накладывают жестких требований, а выступают в качестве рекомендаций к структуре, именованиям и содержанию разделов технического задания. По инициативе организации, выступающей заказчиком или разработчиком технической документации, состав разделов ТЗ может быть расширен или наоборот, часть разделов может быть исключена.

Полный текст и содержание разделов технического задания на ПО, разрабатываемое в рамках выпускной квалификационной работы, приведено в Приложении В.

## 2.4 Варианты использования

Варианты использования (ВИ) – описание последовательности взаимодействия системы и внешнего действующего лица, в результате которого может быть получен конкретный результат.

Для представления вариантов использования может быть применено формальное описание или диаграмма, например в нотации UML.

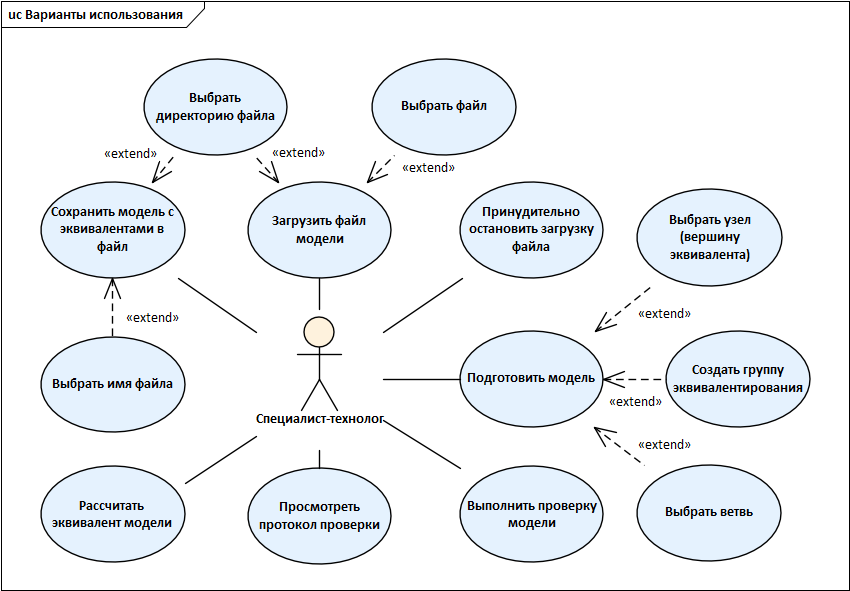


Рисунок 2.5 – Диаграмма вариантов использования

Примечание: для всех ВИ разрабатываемого ПО в качестве акторов является специалист-технолог; в качества стейкхолдеров – Служба электрических режимов (СЭР) и филиал АО «СО ЕЭС».

Таблица 2.1 – ВИ «Загрузить файл модели»

|  |  |
| --- | --- |
| Название | Загрузить файл модели |
| Цель | Загрузить данные о модели энергосистемы для дальнейшей подготовки ее элементов к эквивалентированию |
| Предварительные условия | Система запущена.  Система не осуществляет расчет.  Система не осуществляет загрузку другого файла модели.  Нет открытых диалоговых окон. |
| Основной сценарий | 1. Пользователь инициирует загрузку файла  2. Открывается системный диалог загрузки файла  3. Пользователь выбирает директорию и файл, подтверждает выбор.  4. Закрывается системный диалог загрузки файла.  5. Система загружает файл исходных данных в рабочую область. Конец |
| Альтернативный сценарий №1 | На шаге 3 Основного сценария происходит ошибка загрузки файла (некорректный формат или т.п.).  5. Система информирует Пользователь об ошибке. Конец |
| Альтернативный сценарий №2 | На шаге 3 Основного сценария Пользователь отказался от выбора файла.  4. Системный диалог загрузки файла закрывается. Конец. |

Таблица 2.3 – ВИ «Принудительно остановить загрузку файла»

|  |  |
| --- | --- |
| Название | Принудительно остановить загрузку файла |
| Цель | Досрочно остановить загрузку файла модели |
| Предварительные условия | Выполняется загрузка файла согласно ВИ «Загрузить файл модели» |
| Основной сценарий | 1. Пользователь инициирует остановку загрузки файла  2. Открывается контекстный диалог подтверждения («да» / «нет»)  3. Если «да», то Система останавливает загрузку. Конец  4. Если «нет», то Система продолжает загрузку. Конец |

Таблица 2.3 – ВИ «Выбрать узел-вершину эквивалента»

|  |  |
| --- | --- |
| Название | Выбрать узел (вершину эквивалента) |
| Цель | Определить вершину будущего эквивалента |
| Предварительные условия | Выполнен ВИ «Загрузить файл модели» |
| Основной сценарий | 1. Система предоставляет перечень узлов модели  2. Пользователь выбирает узел модели  3. Система фиксирует выбор. Конец |

Таблица 2.4 – ВИ «Создать группу эквивалентирования»

|  |  |
| --- | --- |
| Название | Создать группу эквивалентирования |
| Цель | Создать группы эквивалентирования для наполнения ее ветвями |
| Предварительные условия | Выполнен ВИ «Выбрать узел (вершину эквивалента)» |
| Основной сценарий | 1. Система предоставляет перечень узлов (вершин модели). По умолчанию выбран первый узел из перечня.  2. Система предоставляет перечень групп эквивалентирования. По умолчанию перечень пуст.  3. Пользователь выбирает узел (вершину модели)  4. Пользователь инициирует создание группы.  5. Система добавляет группу в перечень. Конец. |

Таблица 2.5 – ВИ «Выбрать ветвь»

|  |  |
| --- | --- |
| Название | Выбрать ветвь |
| Цель | Определить объекты модели для последующего их эквивалентирования |
| Предварительные условия | Выполнен ВИ «Создать группу эквивалентирования» |
| Основной сценарий | 1. Система предоставляет перечень узлов (вершин модели). По умолчанию выбран первый узел из перечня.  2. Система предоставляет перечень групп эквивалентирования. По умолчанию выбрана первая группа из перечня.  3.Система предоставляет перечень ветвей модели.  4. Пользователь выбирает ветвь.  5. Система фиксирует выбор. Конец. |

Таблица 2.6 – ВИ «Выполнить проверку модели»

|  |  |
| --- | --- |
| Название | Выполнить проверку модели |
| Цель | Проверить возможность подготовленной модели к эквивалентированию |
| Предварительные условия | Выполнен ВИ «Подготовить модель» |
| Основной сценарий | 1. Пользователь инициирует проверку модели  2. Система осуществляет проверку модели  3. Система записывает (при наличии) ошибки в протокол. Конец. |

Таблица 2.7 – ВИ «Просмотреть протокол проверки»

|  |  |
| --- | --- |
| Название | Просмотреть протокол проверки |
| Цель | Проверить наличия ошибок по результатам проверки в ходе подготовки модели к эквивалентированию |
| Предварительные условия | Выполнен ВИ «Выполнить проверку модели» |
| Основной сценарий | 1. Пользователь инициирует просмотр протокола  2. Система предоставляет перечень (при наличии) ошибок. Конец. |

Таблица 2.8 – ВИ «Рассчитать эквивалент»

|  |  |
| --- | --- |
| Название | Рассчитать эквивалент |
| Цель | Получить эквиваленты выбранных объектов |
| Предварительные условия | Выполнен ВИ «Выполнить проверку модели». В протоколе (ВИ «Просмотреть протокол проверки) отсутствуют ошибки. |
| Основной сценарий | 1. Пользователь инициирует расчет эквивалента  2. Система осуществляет эквивалентирование  4. Система создает эквивалентные объекты, полученные по результатам расчета. Конец |

Таблица 2.9 – ВИ «Сохранить модель с эквивалентами в файл»

|  |  |
| --- | --- |
| Название | Сохранить модель с эквивалентами в файл |
| Цель | Применение модели с эквивалентами для расчетов установившихся режимов в ПК «RastrWin3» |
| Предварительные условия | Выполнен ВИ «Рассчитать эквивалент модели» |
| Основной сценарий | 1. Пользователь инициирует сохранение полученной модели с эквивалентами в файл  2. Открывается контекстный диалог сохранения файла  3. Пользователь выбирает директорию и указывает имя файла, подтверждает выбор.  4. Система сохраняет модель с эквивалентами в файл. Конец. |
| Альтернативный сценарий №1 | На шаге 3 Основного сценария Пользователь отказался от сохранения файла.  3. Закрывается системный диалог сохранения файла. Конец |

## 2.5 Диаграмма компонентов

Диаграмма компонентов – элемент представления архитектурного устройства программы, позволяющий определить зависимости между модулями системы.

Разрабатываемое программное обеспечение ПО «Эквивалент СЭС» (далее – Система) представлена в виде настольного приложения. Данное проектное решение принято в угоду скорости разработки и представления минимально жизнеспособного продукта в ограниченные строки.

В случае положительной оценки продукта, наработки ПО «Эквивалент СЭС» могут быть интегрированы в ПО «Интеграции», которая так же является настольным приложением. Для этого в основу архитектуры Системы положены принципы модульного подхода.

В качестве архитектурного паттерна выбран MVVM (Model‑View‑ViewModel), позволяющий разделить приложение на три части:

* Model – модель данных;
* View – представление (пользовательский интерфейс);
* ViewModel – модель представления, организующая представления данных на View и обработки изменений в них, например в результате пользовательского взаимодействия.

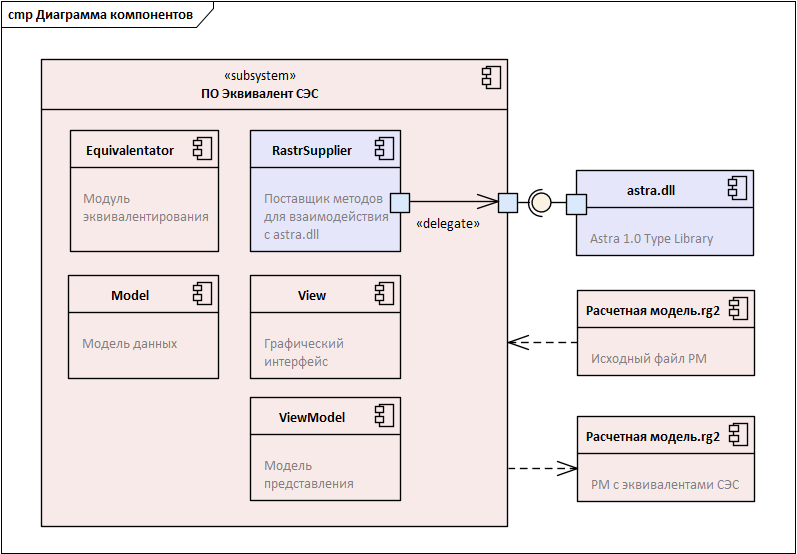


Рисунок 2.6 – Диаграмма компонентов

Среди компонентов Системы могут быть выделены:

* Model – модель данных (элементы модели энергосистемы и графа);
* RastrSupplier – поставщик методов для взаимодействия с astra.dll;
* ViewModel – модель представления;
* Equivalentator – модуль эквивалентирования;
* View – графический интерфейс.

RastrSupplier определяет методы загрузки и сохранения модели энергосистемы ПК «RastrWin3» посредством COM-интерфейса, реализуемого библиотекой astra.dll.

В Equivalentator реализованы методы поиска пути между вершинами графа алгоритмом Дейкстры и эквивалентирования элементов коллекторной сети СЭС.

## 2.6 Макет пользовательского интерфейса

К пользовательскому интерфейсу (UI – User Interface) относятся все компоненты интерактивной системы, которые предоставляют пользователю информацию и являются инструментами взаимодействия с системой для выполнения задач.

Одним из основных свойств интерфейса является – эргономичность (usability), являющейся проводником для положительного пользовательского опыта (UX – User eXpirience) при взаимодействии с системой.

При этом практически невозможно задокументировать требования к интерфейсу в виде текстовых описаний, для однозначной их интерпретации и проверки в ходе приема-сдаточных работ. Зачастую причиной этого является субъективное представления заказчика и исполнителя о визуальном представлении UI, а UX вовсе может быть основан на существующем опыты взаимодействия с прочими системами из ИТ-инфраструктуры заказчика.

Таким образом разработка интерфейса обычно осуществляются в несколько этапов. В ходе которых интерфейс системы проходит от идеи и эскизирования, до разработки прототипа и итоговой его реализации.

Интерфейс ПО «Эквивалент СЭС» представлен одной формой и несколькими рабочими областями (вкладками).

На рисунке 2.7 представлена главная форма с вкладкой «Узлы эквивалентирования». Вверху формы находится «Меню» с элементами «Файл», «Расчет» и «Помощь». Элементы предоставляют доступ к загрузке и сохранению файлов модели энергосистемы («Файл»), инициализации и принудительной остановки расчета («Расчет»), вызова справки и описания программы («Помощь»).

Вкладка «Узлы эквивалентирования» предоставляет доступ к выбору узлов модели, определяющих вершины будущих эквивалентов. Для этого рабочая область разделена на две части, в левой части находится перечень узлов исходной модели, в правой части представлен перечень выбранных узлов-вершин. Перечни узлов отображаются в табличном виде.

Вкладка «Группы эквивалентирования» (рисунок 2.8) предоставляет доступ к формированию групп эквивалентирования. В левой верхней части экрана представлен перечень узлов-вершин эквивалентов, ранее выбранных на вкладе «Узлы эквивалентирования». Выбор узла из перечня позволяет создать группу эквивалентирования (правая верхняя часть экрана). А после выбора группы осуществить ее наполнение ветвями, которые буду подвержены эквивалентированию.

Наполнение групп ветвями осуществляется по аналогичному принципу, как и выбор узлов на вкладке «Узлы эквивалентирования». Рабочая область поделена на две части. В левой нижней части представлен перечень ветвей исходной модели, в правой нижней – перечень выбранных ветвей для соответствующей группы и узла-вершины эквивалента. Перечни ветвей отображаются в табличном виде.

При возникающих ошибках в ходе загрузки / сохранения файлов модели, проверки данных, расчета и принудительной отмены в интерфейсе предусмотрен вызов окон сообщений для уведомления и подтверждения действий.

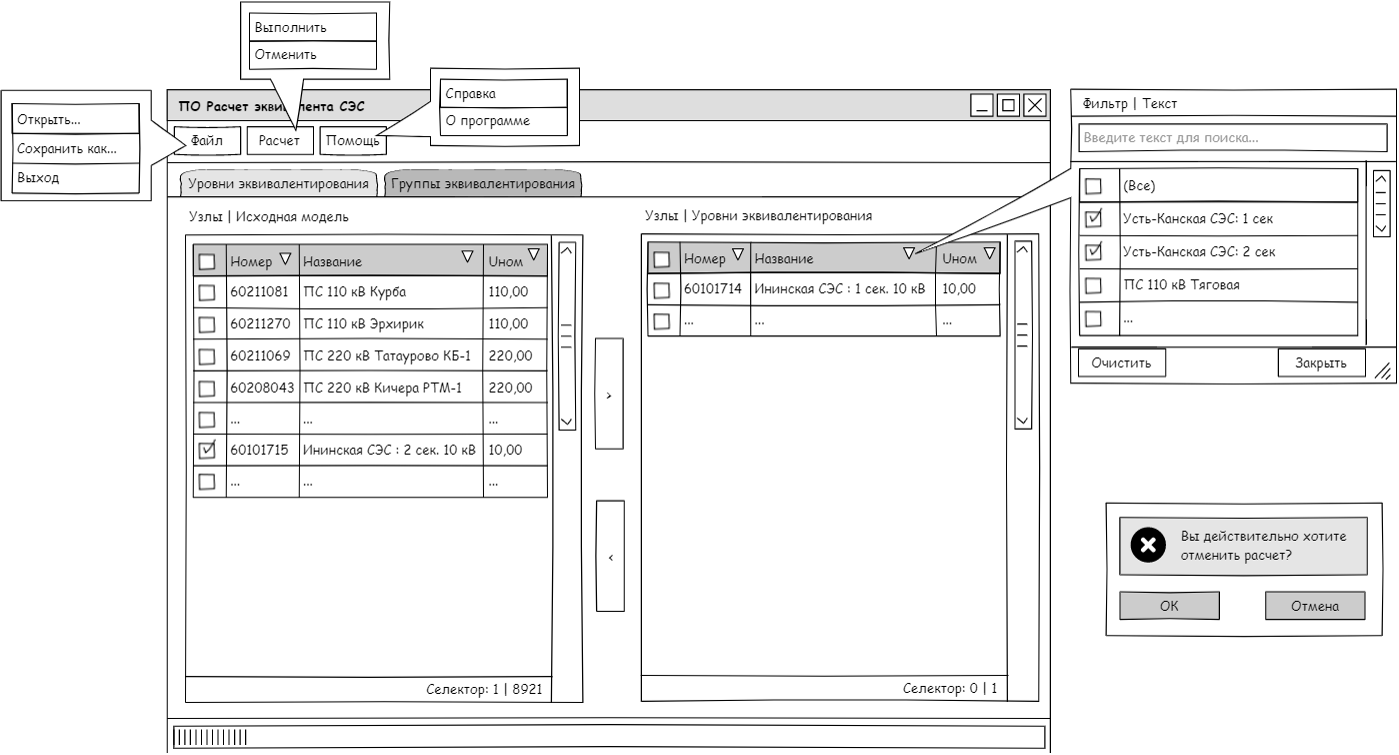


Рисунок 2.7 – Вкладка «Узлы эквивалентирования»

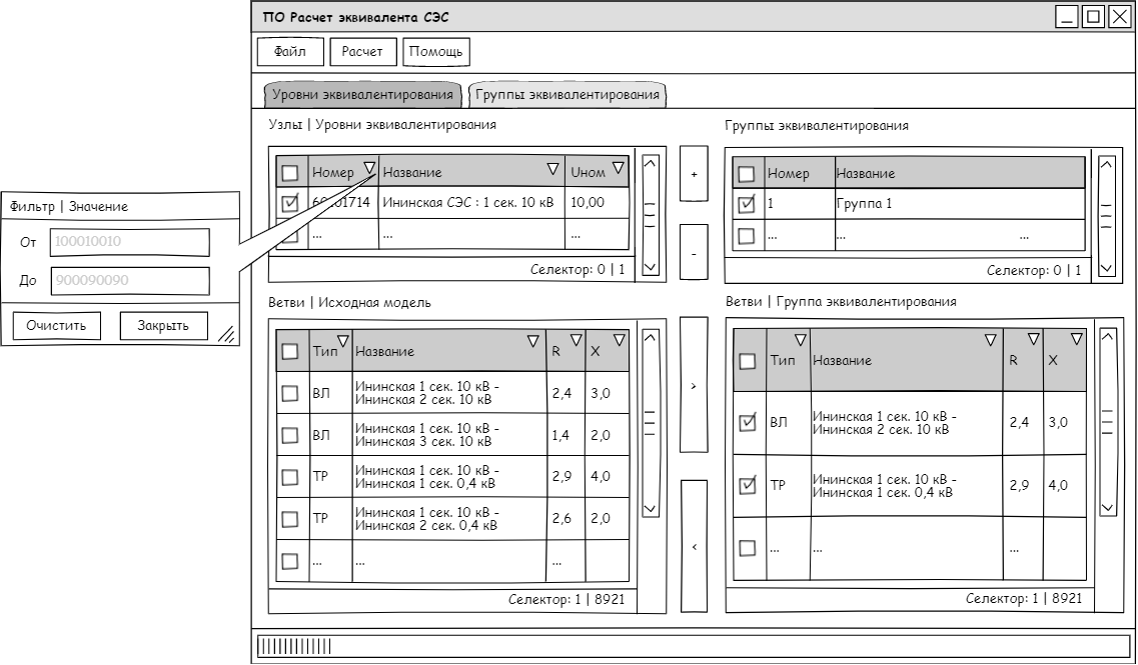


Рисунок 2.8 – Вкладка «Группы эквивалентирования»

## 2.7 Выводы по разделу

В результате выполнения раздела были проработаны механизмы интеграции разрабатываемого программного обеспечения, как с позиции существующих деловых процессов, так и со стороны имеющейся ИТ‑инфраструктуры.

Для дальнейшей разработки и концептуального проектирования был выбран вариант «Эквивалентирование расчетной модели». Данный вариант предоставляет MVP системы в условиях ограниченности временных ресурсов. Остальные интеграционные механизмы представляют вариант развития системы или модификацию смежных.

Краткая сравнительная характеристика вариантов интеграции приведена в таблице 2.7.

Таблица 2.7 – Сравнительная характеристика вариантов интеграции

|  |  |  |  |
| --- | --- | --- | --- |
| Критерий | Вариант 1. Эквивалентирование РМ | Вариант 2. Эквивалентирование ИМ | Вариант 3. Эквивалентирование РМ по данным ИМ |
| Работа с ИМ (следование тенденциям развития АСДУ) | ◯ | ● | ◗ |
| Независимость от подхода моделирования СЭС в КМ | ● | ◯ | ◗ |
| Возможность самостоятельной реализации | ● | ◯ | ● |
| Сроки реализации | ● | ◯ | ◯ |
| Отсутствие издержек на сопровождение ПО | ◯ | ● | ◯ |

Разработан алгоритм эквивалентирования (Приложение Б). Для прикладной реализации расчета эквивалента методом суммарных эквивалентных потерь применен поиск кратчайшего пути в графе алгоритмом Дейкстры.

Осуществлен инжиниринг требований к разрабатываемому программному обеспечению. Техническое задание представлено в Приложении В.

Для проработки требований к ПО «Эквивалент СЭС» определены ВИ, в последствие представленные в формате диаграммы и текстовых описаний.

Выбрана и обоснована архитектура системы. ПО «Эквивалент СЭС» будет выполнено в формате настольного приложения. Модульность архитектуры обеспечивается применением шаблона проектирования MVVM. Для визуализации выбранных архитектурных решений оформлена диаграмма компонентов.

В заключении разработаны макеты будущего интерфейса системы. Интерфейс представлен одной формой с разделением рабочих областей посредством вкладки, предоставляющих различный функционал по подготовке моделей к эквивалентированию.

# 3. Программная реализация

## 3.1 Выбор инструментов

Архитектурный стиль MVVM, выбранный ранее в качестве шаблона проектирования, применяется в разработке настольных приложений под ОС Windows, а также мобильных устройств под управлением Android и iOS.

Целевой операционной системой для разработки ПО «Эквивалент СЭС» является Windows, которая является основной на рабочих станциях специалистов АО «СО ЕЭС».

На текущий момент в Windows существует два основных API для работы приложений – Windows API (Win32) и Windows Runtime (WinRT). Программный интерфейс Win32 на текущий момент не развивается. Современная разработка настольных приложений Windows ведется посредством взаимодействия с WinRT. WinRT предоставляет COM интерфейс доступ к методам которого может быть получен посредством практически любого современного языка программирования C++, Rust, Python, С# и т.п.

При этом среди приеденных языков С# имеет ряд преимуществ. Так, данный язык программирования:

* развивается той же командой разработчиков, что занимается Windows;
* использует программную платформу .NET являющейся кроссплатформенной, что позволяет исполнять приложения, например, на ОС семейства Linux;
* имеет большой перечень UI платформ для разработки под Windows, например WPF (Windows Presentation Foundation).

Технология WPF является частью экосистемы .NET, графические интерфейсы в которой разрабатываются с применением языка разметки XAML (eXtensible Application Markup Language).

Использование WPF предоставляет ряд преимуществ:

* удобство разработки адаптивно масштабируемых интерфейсов;
* применение аппаратного ускорения для отрисовки графики.

В результате для дальнейшей разработки выбран язык программирования C# 10, программная платформа .NET 6 и технология WPF последних стабильных и актуальных (поддерживаемых) версий.

Наиболее популярными и удобными средами разработки (IDE – Integrated Development Environment) для языка программирования C# являются Visual Studio, Project Rider и VS Code.

Project Rider от компании JetBrains не имеет Community версии для свободного использования, VS Code в большей степени представляет многофункциональный редактор кода, требующий установки множества дополнительных плагинов, чтобы иметь возможность разработки C# (.NET) приложений.

Итого выбрана среда разработки Visual Studio.

Для версионирования кода применяется распределенная система управления версиями git, которая де-факто является стандартом в ИТ-отрасли. Использование git позволяет подключать удаленные сервера для публикации и хранения кода, например, на GitHub или GitLab.

## 3.1 Model (модель данных)

Для описания сущностей элементов энергосистемы, хранящихся в табличном виде в модели ПК «RastrWin3», в разрабатываемой системы были реализованы соответствующие им классы: Node, Branch и Generator.

Классы содержат минимальный и достаточный набор атрибутов, сопоставленных с элементами модели RastrWin3, для их однозначной идентификации и реализации бизнес-логики эквивалентирования.

Описание классов элементов сети приведено в Приложении Г.

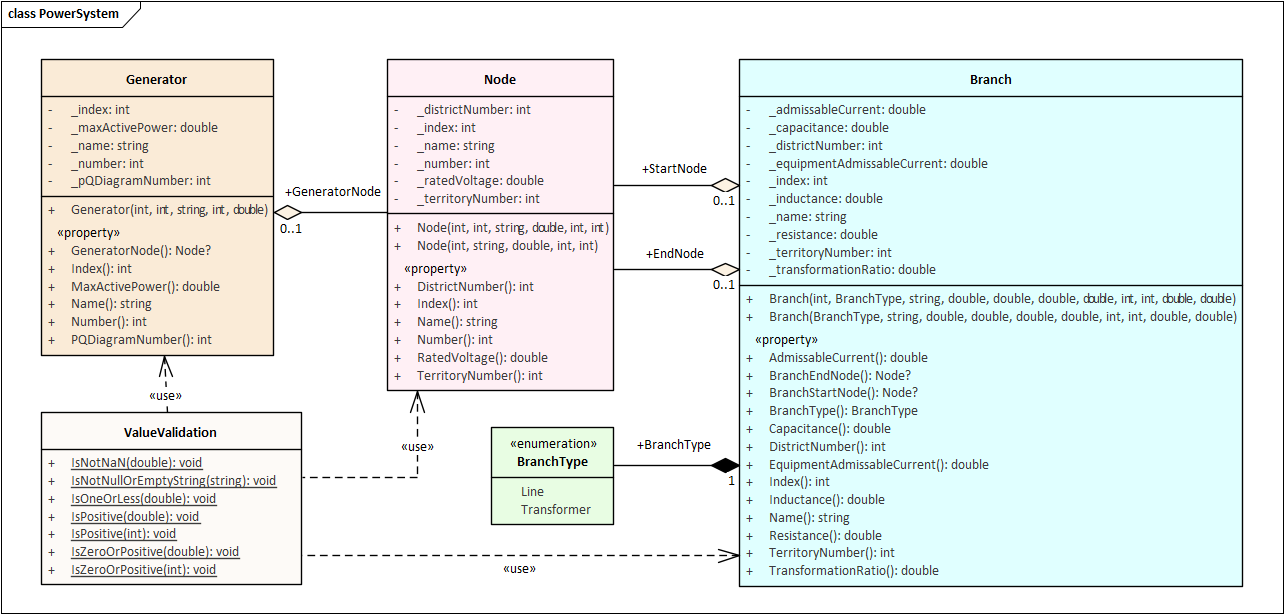


Рисунок 3.1 – Диаграмма классов модели элементов энергосистемы

Для реализации бизнес-логики эквивалентирования с помощью алгоритма Дейкстры были реализованы обобщенные классы: Graph, GraphVertex, GraphEdge, GraphVertexInfo и Dijkstra.

Обобщение типов (generics) позволяет разрабатывать программную логику, которая будет одинаково работать с различными типами данных с сохранением строгой типизации языка.

Описание классов элементов структуры графа приведено в Приложении Г.

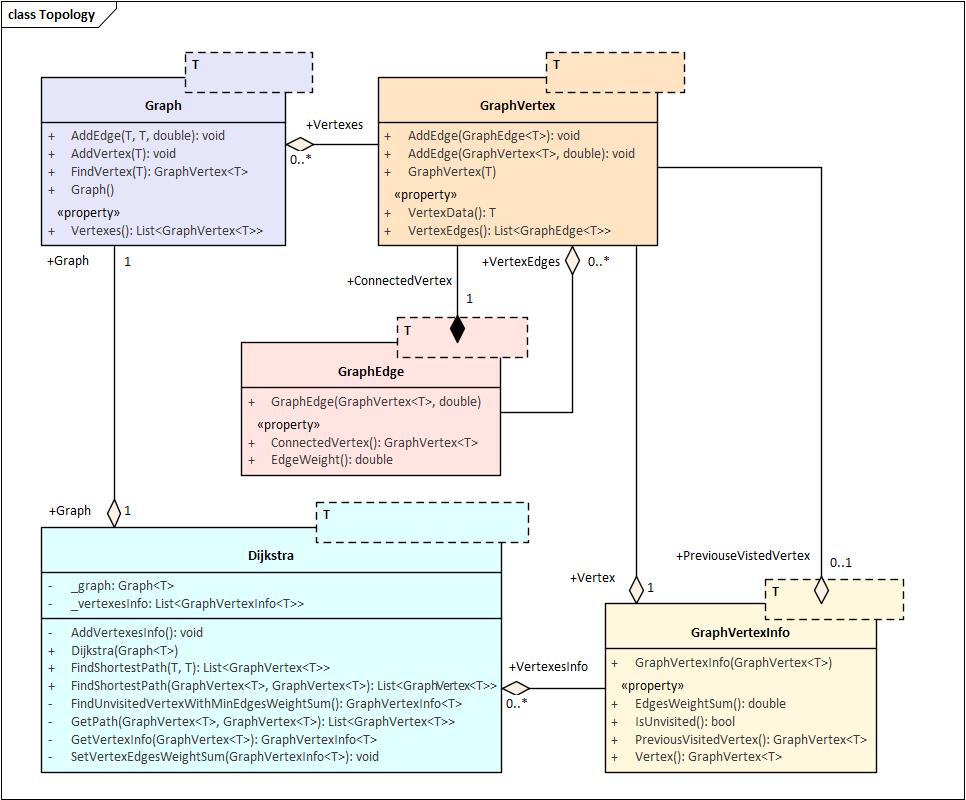


Рисунок 3.2 – Диаграмма обобщенных классов модели графа

При этом в рамках предметной области вершинами графа (Vertex) выступают экземпляры класса Node, а ребрами – экземпляры Branch (их параметры).

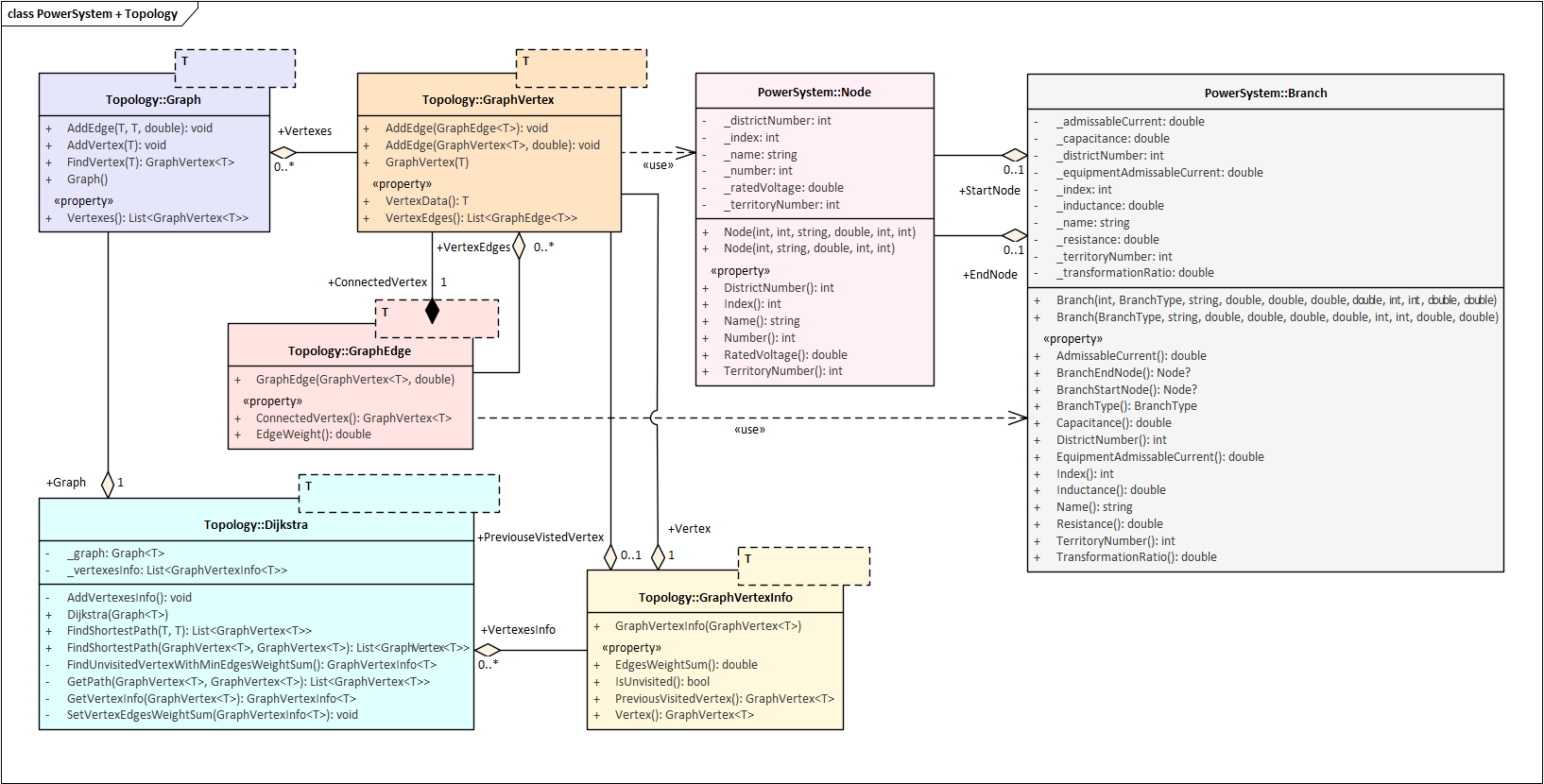


Рисунок 3.3 – Диаграмма классов модели элементов энергосистемы и обобщенных классов модели графа

## 3.3 RastrSupplier

Динамическая библиотека astra.dll поставляется совместно с RastrWin3 и предоставляет доступ к функциям и методам программного комплекса.

Для вызова функций и методов непосредственно из программного кода необходимо создать ссылку на COM модель библиотеки в проекте.

Далее необходимо подключить пространство имен ASTRALib и создать экземпляр класса Rastr.

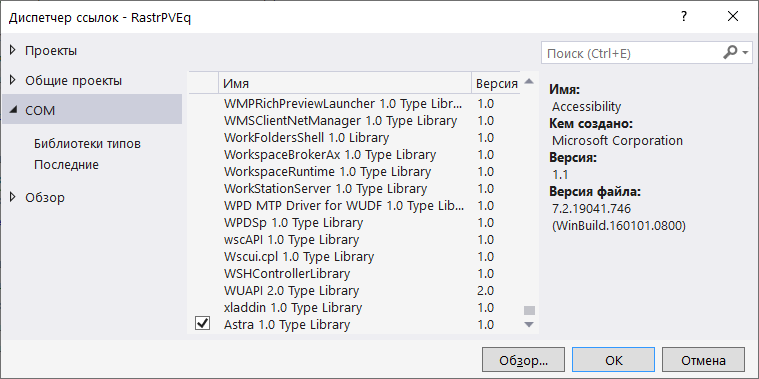


Рисунок 3.4 – Диспетчер ссылок проекта

Для взаимодействия с моделью энергосистемы используются методы, реализуемые интерфейсами ITables, ITable, ICols и ICol. Описание реализуемых методов приведены в Руководстве пользователя [], либо могут быть инспектированы посредством декомпиляции astra.dll, например, через свободно распространяемую утилиту ILSpy [].

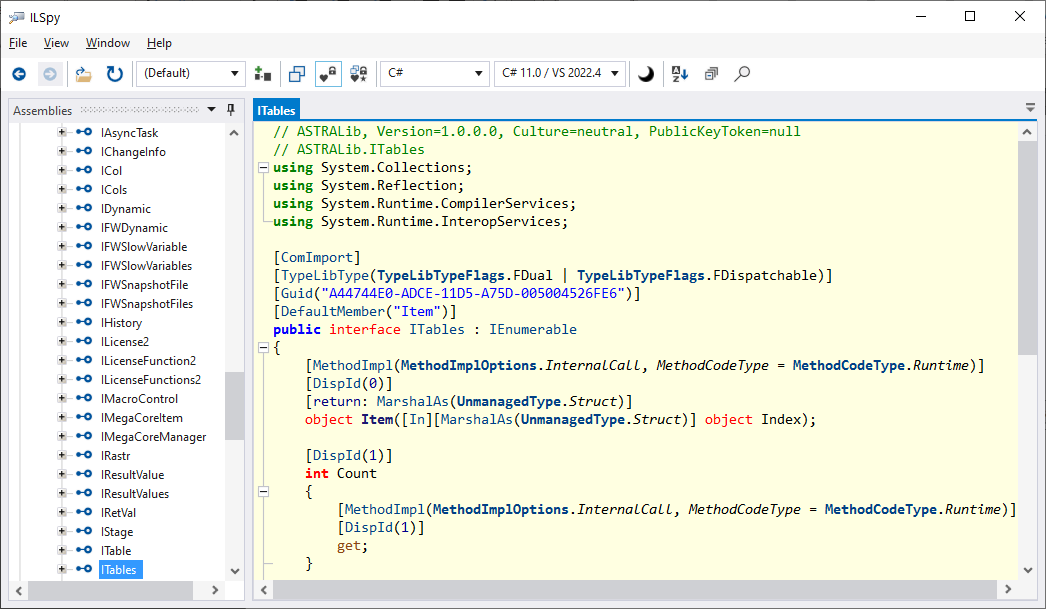


Рисунок 3.2 – Результат декомпиляции astra.dll

В результате, для взаимодействия с моделью энергосистемы ПК «RastrWin3» были разработаны статические классы RastrProvider и RastrProviderAsync, реализующие синхронные и асинхронные методы взаимодействия с astra.dll, посредством предоставляемых интерфейсами ITables, ITable, ICols и ICol методов.

Асинхронные методы, выполняемые параллельно, с помощью библиотеки параллельных задач TPL (Task Parallel Library) позволяют сократить время загрузки модели энергосистемы в приложение.

Также в рамках разработки модуля взаимодействия с astra.dll определен вспомогательный статический класс RastrConstantNames со строковыми ключами таблиц и параметров элементов RastrWin3.

Детальное описание классов RastrSupplier и реализуемых ими методов приведено в Приложении Г.

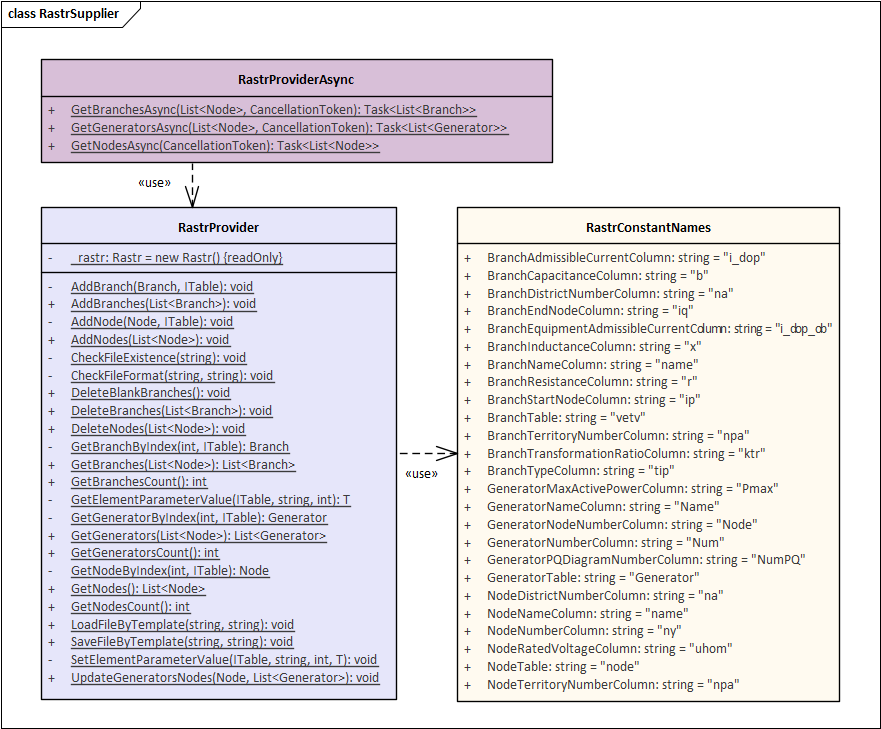


Рисунок 3.3 – Диаграмма классов RastrSupplier

## 3.4 ViewModel (модель представления)

Для реализации модели представления в рамках архитектурного стиля MVVM используется официальная библиотека от Microsoft, распространяемая в формате NuGet пакета – CommunityToolkit.Mvvm [].

Пакет ориентирован на стандарт .NET, поэтому его можно использовать на любой платформе приложений: UWP, WinForms, WPF, Xamarin, Uno, а также в любой среде выполнения: .NET Native, .NET Core, .NET Framework или Mono.

Библиотека обеспечивает встраивание интерфейса INotifyPropertyChanged, и реализацию ObservableObject, ObservableRecipient, RelayCommand в пользовательских классах, их полях и методах посредством указания соответствующих атрибутов.

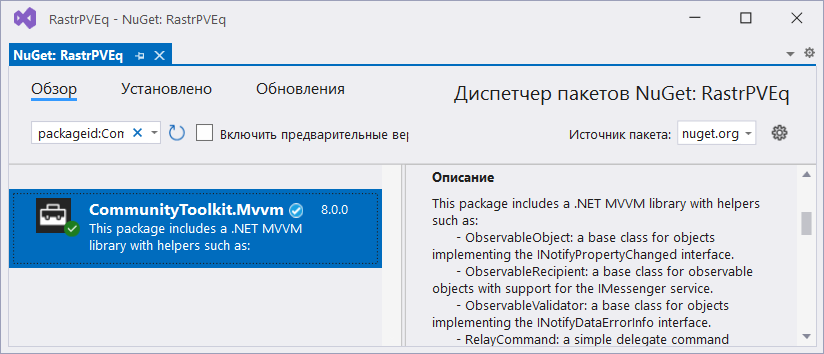


Рисунок 3.4 – CommunityToolkit.Mvvm в диспетчере NuGet пакетов

Листинг кода 3.1 – Фрагмент кода с определением атрибутов INotifyPropertyChanged, ObservableProperty и RelayCommand

**using** CommunityToolkit.Mvvm.ComponentModel;

**using** CommunityToolkit.Mvvm.Input;

**namespace** RastrPVEq.ViewModels

{

*/// <summary>*

*/// Main Window View Model class*

*/// </summary>*

[INotifyPropertyChanged]

**public** **partial** **class** MainWindowViewModel

{

*/// <summary>*

*/// Nodes*

*/// </summary>*

[ObservableProperty]

**private** ObservableCollection<Node> \_nodes;

...

*/// <summary>*

*/// Download file command*

*/// </summary>*

[RelayCommand]

**private** **async** **void** DownloadFile()

{

...

}

}

В рамках реализации модели представления разработаны классы MainWindowViewModel, EquivalenceNodeViewModel и EquivalenceGroupViewModel.

Детальное описание классов модели представления приведено в Приложении Г.

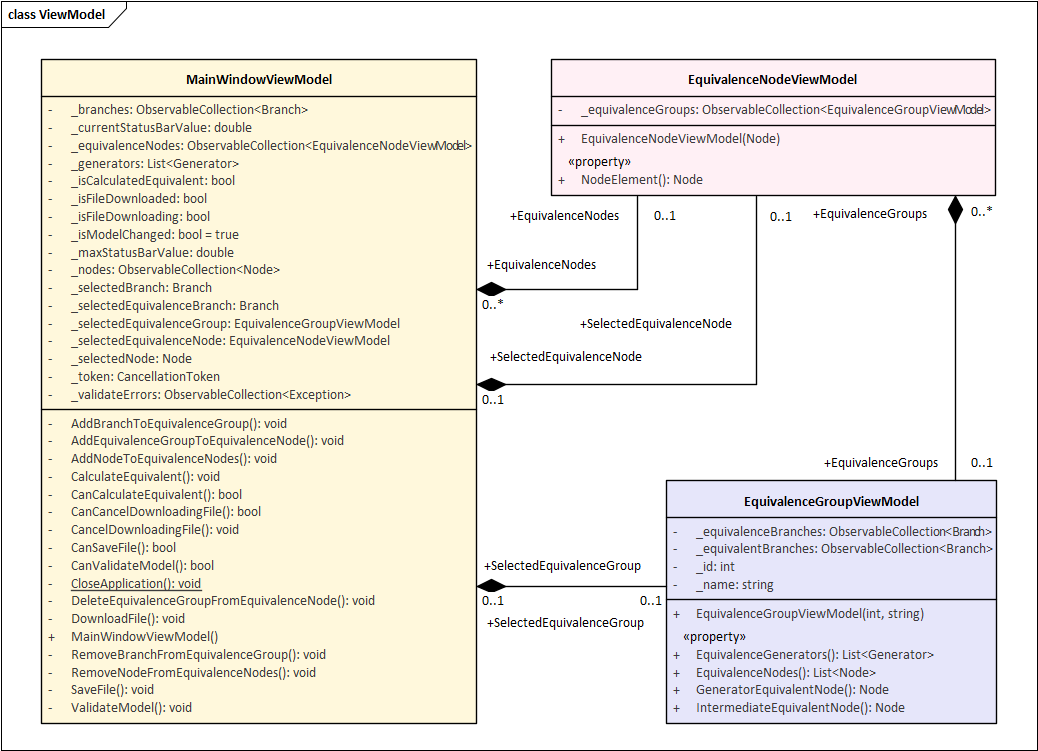


Рисунок 3.5 – Диаграмма классов модели представления

## 3.5 Equivalentator

Статический класс Equivalentator предоставляет набор методов для валидации модели, перед ее эквивалентированием, и соответственно расчета эквивалента согласно алгоритму эквивалентирования (Приложение Б).

Детальное описание класса Equivalentator и реализуемых им методов приведено в Приложении Г.

## 3.6 View (представление)

View представлен частичным (partial) классом – MainWindow.xaml, содержащий код C# (в MainWindow.xaml.cs) и непосредственно разметку XAML.

Листинг кода 3.2 – Фрагмент XAML разметки

**<Window** x:Class="RastrPVEq.MainWindow"

...

**<Window.DataContext>**

**<vm:MainWindowViewModel/>**

**</Window.DataContext>**

...

**<DockPanel>**

...

**</DockPanel>**

**</Window>**

Для стилизации интерфейса используется Material Design In XAML Toolkit – его NuGet пакет MaterialDesignThemes [].

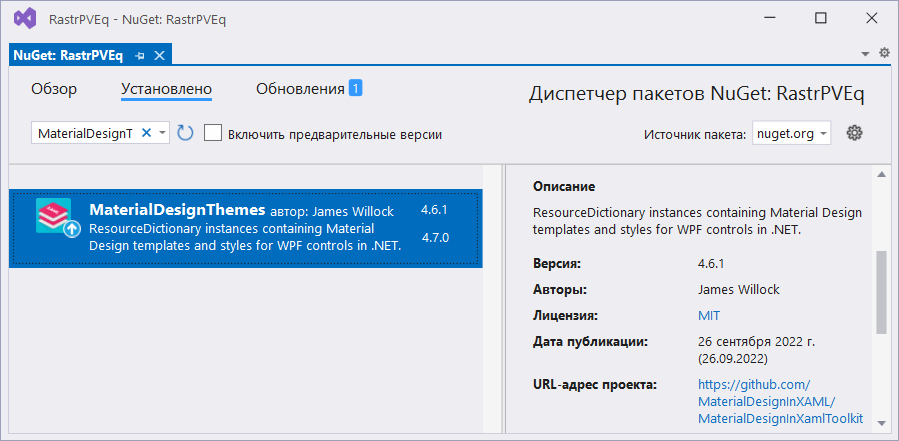


Рисунок 3.6 – MaterialDesignThemes в диспетчере NuGet пакетов

При этом, для применения стиля MaterialDesignThemes вносятся изменения в App.xaml, содержащий ссылки на все стилистические ресурсы, используемые в приложении.

Также используется пользовательский элемент управления, модифицирующий стандартный DataGridView, предназначенный для отображения информации в табличном виде, и реализующий в последнем возможность фильтрации данных.

Пользовательский элемент реализован на основе FilterDataGrid. При этом сам элемент может быть получен посредством установки NuGet пакета.

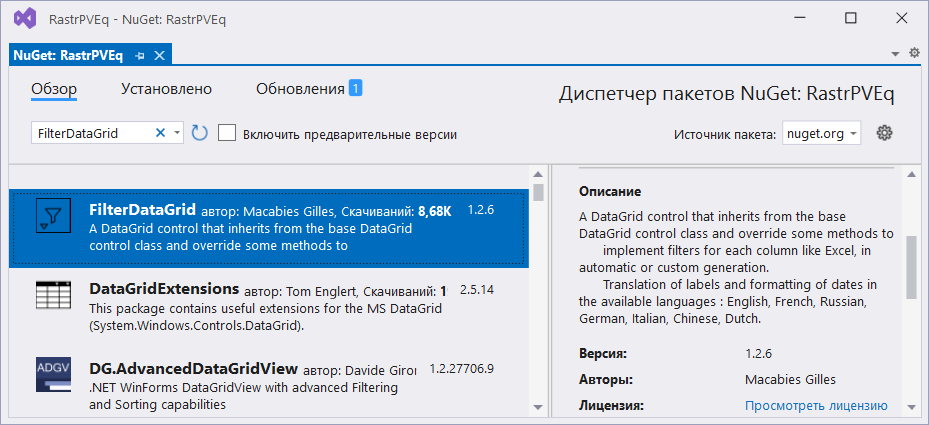


Рисунок 3.7 – FilterDataGrid.Net в диспетчере NuGet пакетов

На основании макетов интерфейса и описанного их функционального назначения (п. 2.8) разработаны главная форма приложения и вкладки:

* «Вершины эквивалентов» для выбора узлов модели (вершин эквивалентов);
* «Группы эквивалентов» для создания групп эквивалентирования и выбора ветвей;
* «Протокол» для отображения результатов проверки модели перед эквивалентированием;
* «Результат» для отображения результатов эквивалентирования.

Реализация интерфейса приведена на рисунках 3.8 – 3.Х.

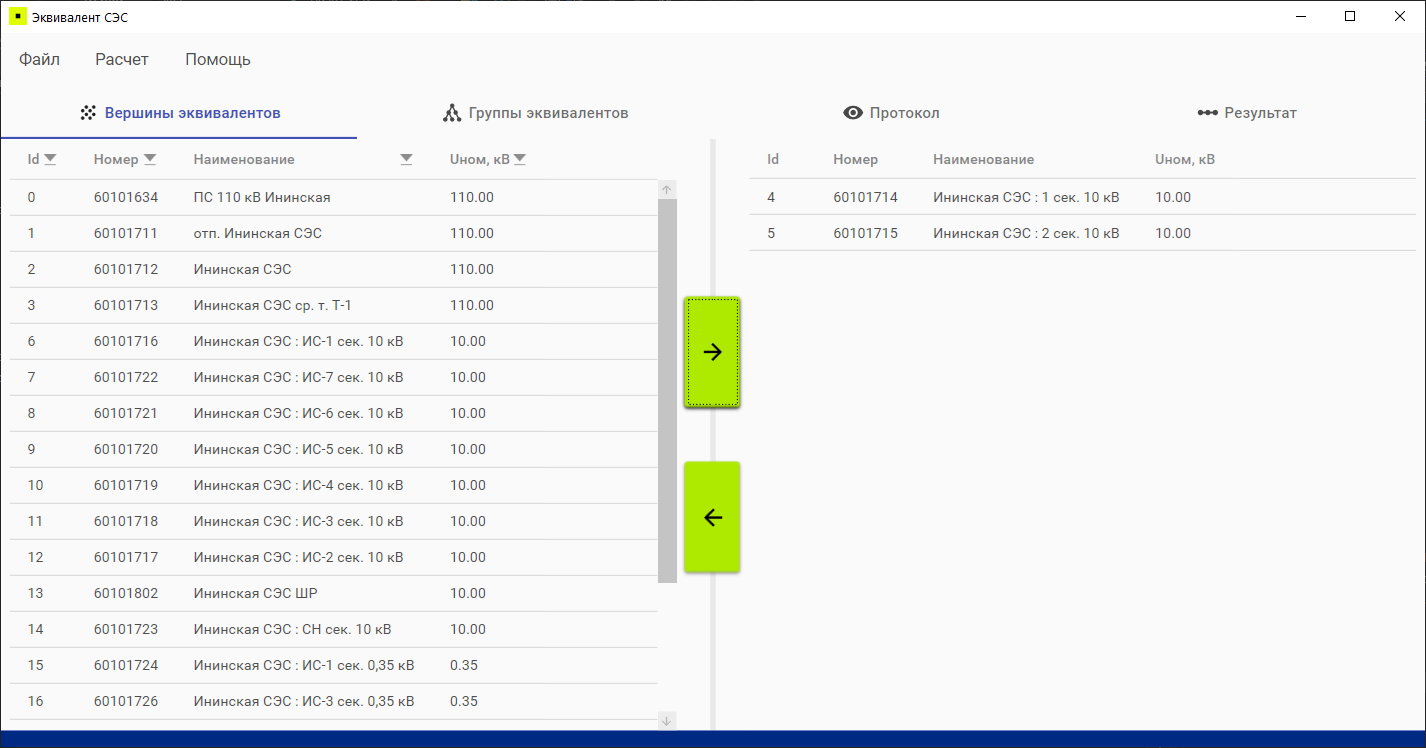


Рисунок 3.8 – Вкладка «Вершины эквивалентов»

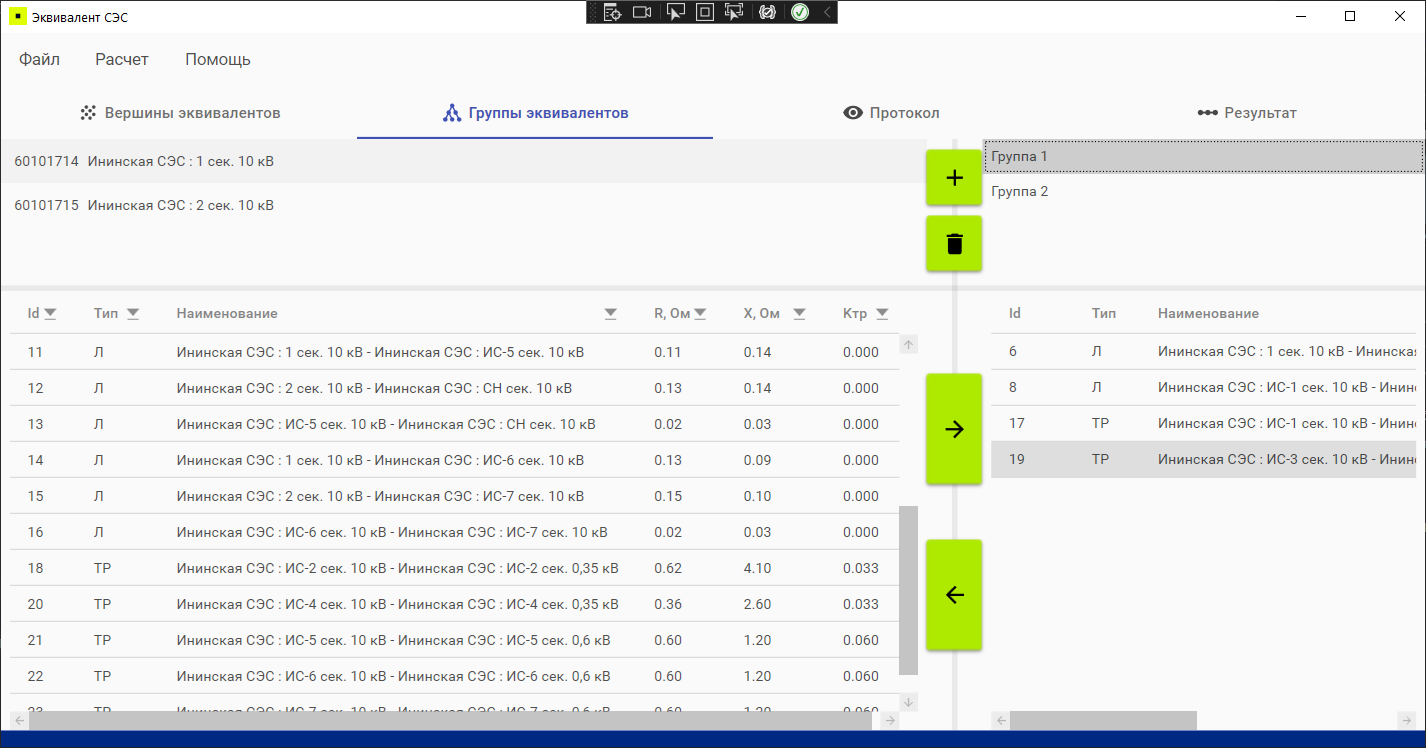


Рисунок 3.9 – Вкладка «Группы эквивалентов»

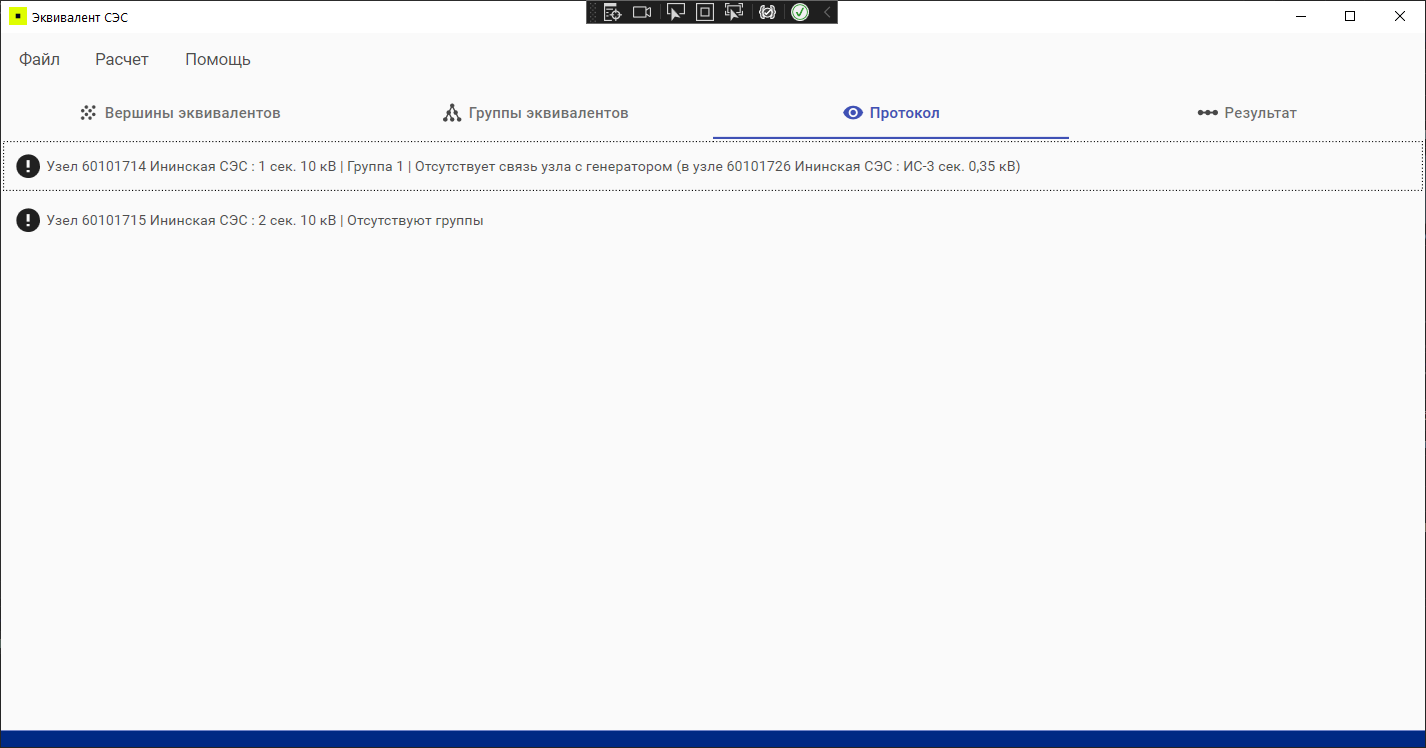


Рисунок 3.10 – Вкладка «Протокол»

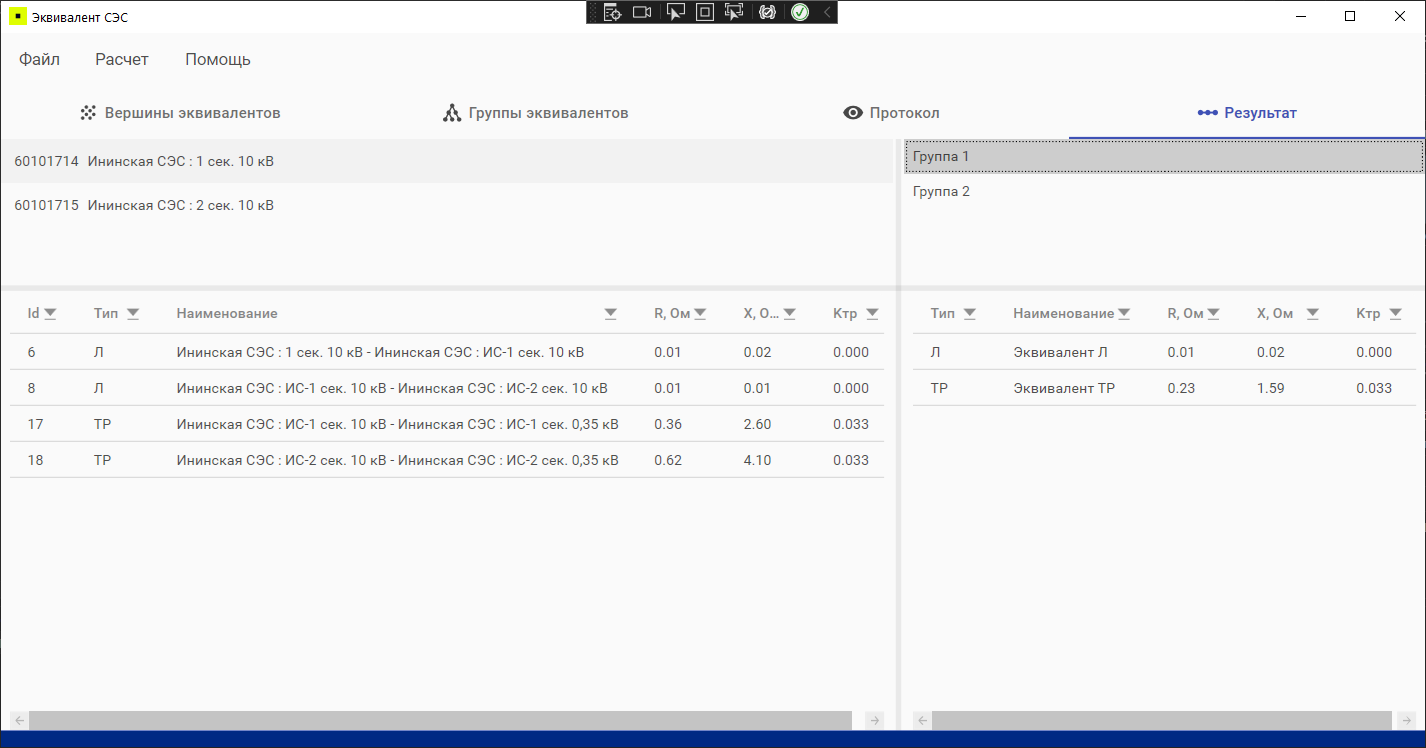


Рисунок 3.11 – Вкладка «Результат»

## 3.7 Выводы по разделу

В результате выполнения раздела был обоснован и осуществлен выбор основных инструментов разработки.

В качестве архитектурного стиля определен шаблон MVVM, подразумевающий разделение приложение на три укрупненных компонента: Model, ViewModel и View.

Программная реализация осуществляется на языке программирования C# и платформы .NET, а также UI технологии WPF.

Модели элементов энергосистемы ПК «RastrWin3» определены в классах из пространства имен PowerSystem, модель графа – классами из Topology.

Для RastrSupplier определены статические классы, представляющие синхронные и асинхронные реализации методов взаимодействия с моделью ПК «RastrWin3» через COM-интерфейсы библиотеки astra.dll.

Шаблон MVVM в ViewModel обеспечивается применением библиотеки CommunityToolkit.Mvvm, предоставляющей удобный способ реализации интерфейса INotifyPropertyChanged в пользовательских классах.

Реализован Equivalentator – статический класс методов проверки модели и расчета эквивалента.

Взаимосвязь реализованных компонентов и их классов представлена посредством UML диаграмм, а их описание приведено в табличном виде.

В заключение разработан графический интерфейс приложения.

# 4. Тестирование

# СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

1. Генеральная схема размещения объектов электроэнергетики на период до 2035 года от 17.10.2009 № 823;
2. Force W. R. E. M. T. Solar Photovoltaic Power Plant Modeling and Validation Guideline. 2019;
3. Паспорт Программы инновационного развития АО «СО ЕЭС» на 2017‑2021 годы и на перспективу до 2025 года от 15.10.2021;
4. IEC 61970-552:2016 Интерфейс прикладных программ систем энергетического менеджмента (EMS-API). Часть 552. Обмен моделями в формате CIMXML (Energy management system application program interface (EMS-API) – Part 552: CIMXML Model exchange format);
5. Регламент взаимодействия ОДУ Сибири и РДУ ОЗ ОДУ Сибири при формировании и внесении изменений в расчетные модели ОЭС Сибири;
6. Регламент взаимодействия структурных подразделений исполнительного аппарата и филиалов АО «СО ЕЭС» ОДУ, филиалов АО «СО ЕЭС» РДУ при актуализации данных информационной модели;
7. Программный комплекс «RastrWin3». Руководство пользователя (коммутационные схемы). ред. от 27.07.2020;
8. ГОСТ Р 58651 Информационная модель «Основные положения»;
9. ГОСТ Р 58651.2-2019 «Базисный профиль информационной модели»;
10. ГОСТ Р 58651.3-2020 «Профиль информационной модели линий электропередачи и электросетевого оборудования напряжением 110‑750 кВ»;
11. ГОСТ Р 58651.4-2020 «Профиль информационной модели генерирующего оборудования»;
12. ГОСТ Р 58651.6-2022 «Профиль информационной модели линий электропередачи и электросетевого оборудования напряжением 0,4‑35 кВ»;
13. ГОСТ 19.201 78 «Единая система программной документации. Техническое задание. Требования к содержанию и оформлению»;
14. ГОСТ 34.602-2020 «Техническое задание на создание автоматизированной системы»;

# Приложение А

Таблица А.1 – Сопоставление атрибутов «Production::ThermalGeneratingUnit» и «Production::PhotoVoltaicUnit»

|  |  |  |  |
| --- | --- | --- | --- |
| Атрибут | | Назначение | Использование в КМ АО «СО ЕЭС» |
| ThermalGeneratingUnit | PhotoVoltaicUnit |
| Определены в отраслевых стандартах | | | |
| governorSCD | governorSCD | Статизм регулятора, % | ✓ |
| highControlLimit |  | Верхний предел автоматического регулирования, МВт | **×** |
| lowControlLimit |  | Нижний предел автоматического регулирования, МВт | **×** |
| lowerRampRate |  | Скорость снижения мощности турбины, МВт/с | **×** |
| maxOperatingP | maxP | Максимум, МВт | ✓ |
| minOperatingP | minP | Минимум, МВт | ✓ |
| nominalP |  | Номинальная активная мощность, МВт | ✓ |
| raiseRampRate |  | Скорость увеличения мощности, МВт/с | **×** |
|  | rf:deadband | Мертвая полоса первичного регулирования, Гц | **×** |
| Расширение профиля АО «СО ЕЭС» | | | |
| baseP |  | Базовая загрузка, МВт | ✓ |
| initialP |  | Начальная мощность, МВт | ✓ |
| longPF |  | Коэффициент долевого участия (длительный) | ✓ |
| maxEconomicP |  | План Pmax | ✓ |
| minEconomicP |  | План Pmin | ✓ |
| modelDetail |  | Детальность модели | ✓ |
| Расширения профиля АО «Монитор Электрик» | | | |
| genOperatingMode |  | Вторичное регулирование частоты и мощности | ✓ |
| normallyInService |  | Состояние | ✓ |

Таблица А.2 – Сопоставление атрибутов «Wires::SynchronouseMachine» и «Wires::PowerElectronicsConnection»

|  |  |  |  |
| --- | --- | --- | --- |
| Атрибут | | Назначение | Использование в КМ АО «СО ЕЭС» |
| Synchronouse Machine | PowerElectronics Connection |
| Определены в отраслевых стандартах | | | |
| ratedPowerFactor | ratedPowerFactor | Номинальный коэффициент мощности | ✓ |
| ratedS | ratedS | Номинальная полная мощность, МВА | ✓ |
| ratedU | ratedU | Номинальное напряжение, кВ | ✓ |
| condenserP |  | Активная мощность, потребляемая в режиме синхронного компенсатора, МВт | ✓ |
| coolantType |  | Тип охлаждающего агента | ✓ |
| earthing |  | Нейтраль заземлена (если присвоено значение «Истина») |  |
| earthingStarPointR |  | Активное сопротивление заземления нейтральной точки генератора, Ом | **×** |
| earthingStarPointX |  | Реактивное сопротивление заземления нейтральной точки генератора,Ом | **×** |
| maxQ | maxQ | Верхний предел по реактивной мощности, Мвар | ✓ |
| maxU |  | Ограничение по максимальному напряжению, кВ | **×** |
| minQ | minQ | Нижний предел по реактивной мощности, Мвар | ✓ |
| minU |  | Ограничение по минимальному напряжению, кВ | **×** |
| operatingMode |  | Текущий режим работы синхронной машины | ✓ |
| r |  | Активное сопротивление статора, Ом | **×** |
| r0 |  | Активное сопротивление нулевой последовательности, Ом | **×** |
| r2 |  | Активное сопротивление обратной последовательности, Ом | **×** |
| type | type | Режимы работы, в которых синхронная машина способна работать / Тип (марка) преобразователя | ✓ |
| x0 |  | Реактивное сопротивление нулевой последовательности, Ом | **×** |
| x2 |  | Реактивное сопротивление обратной последовательности, Ом | **×** |

Продолжение таблицы А.2 – Сопоставление атрибутов «Wires::SynchronouseMachine» и «Wires::PowerElectronicsConnection»

|  |  |  |  |
| --- | --- | --- | --- |
| Атрибут | | Назначение | Использование в КМ АО «СО ЕЭС» |
| Synchronouse Machine | PowerElectronics Connection |
| Расширение профиля АО «СО ЕЭС» | | | |
| p |  | Активная мощность, МВт | ✓ |
| q |  | Реактивная мощность, Мвар | ✓ |
| Расширения профиля АО «Монитор Электрик» | | | |
| slackBusPriority |  | Приоритет балансирующего узла | ✓ |

# Приложение Б

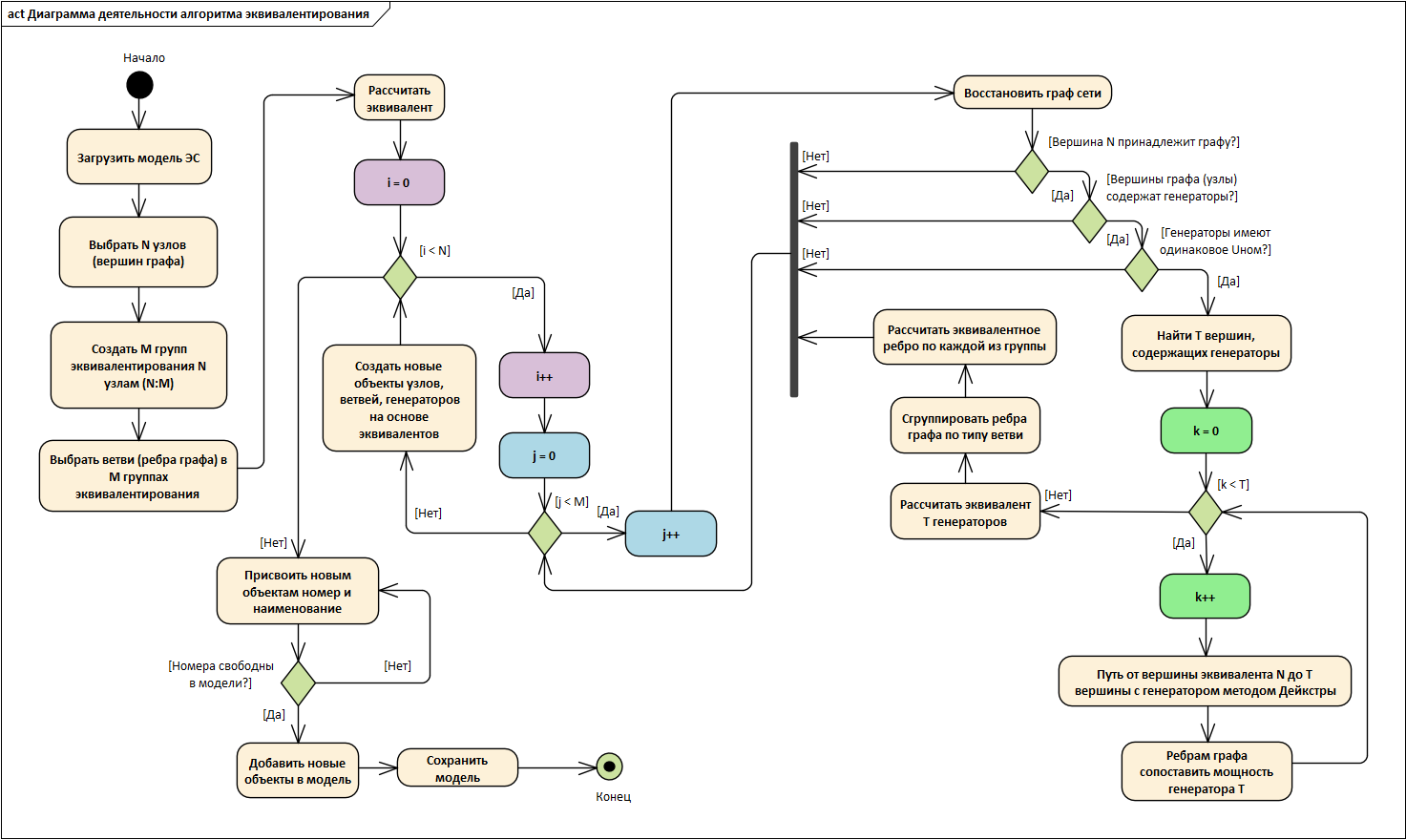


Рисунок Б.1 – Укрупненная диаграмма деятельности алгоритма эквивалентирования

# Приложение В

**ТЕХНИЧЕСКОЕ ЗАДАНИЕ**

НА СОЗДАНИЕ ПРОГРАММЫ ДЛЯ ЭВМ

«ЭКВИВАЛЕНТ СОЛНЕЧНЫХ ЭЛЕКТРОСТАНЦИЙ»

Томск 2022

**1. ОБЩИЕ СВЕДЕНИЯ**

**1.1 Полное наименование системы и ее условное обозначение**

Наименование системы: программа для ЭВМ «Эквивалент солнечных электростанций».

Условное обозначение системы: Система, ПО «Эквивалент СЭС».

**1.2 Сведения об организации заказчика и исполнителя**

Заказчик: Филиал АО «СО ЕЭС» «Объединенное диспетчерское управление энергосистемы Сибири». Адрес: 650000, г. Кемерово, ул. Кузбасская, д. 29.

Исполнитель: Студент гр. О-5КМ01, Чернобров Михаил Евгеньевич Национальный исследовательский Томский политехнический университет. Адрес: 634034, г. Томск, ул. Усова, д. 15Б, ком. 302Б.

**1.3 Основание для разработки системы**

В продолжение научно-исследовательской работы (НИР) в рамках магистерской диссертации на тему «Разработка модели солнечной электростанции для задач регулирования напряжения и реактивной мощности в установившихся режимах электроэнергетической системы» (далее – Диссертация УР).

А также в исполнение требований:

* Федерального государственного образовательного стандарта (ФГОС) высшего образования – магистра по направлению подготовки 09.04.03 «Прикладная информатика» (Приказ Минобрнауки России № 916 от 19 сентября 2017 г.);
* Образовательного стандарта ТПУ по направлению подготовки (СУОС) 09.04.03 «Прикладная информатика» (Приложение 57 к Приказу № 16953 от 21 декабря 2018 г.).

**1.4 Основные понятия, определения и сокращения**

|  |  |
| --- | --- |
| АО «СО ЕЭС» | Акционерное общество «Системный оператор Единой энергетической системы» |
| ЗУД | Защищенный удаленный доступ |
| ИМ | Информационная модель |
| ОДУ | Объединённое диспетчерское управление энергообъединениями |
| ОЗ | Операционная зона |
| ОЭС | Объединенная энергосистема |
| ПК | Программный комплекс |
| ПМИ | Программа и методика испытаний |
| ПО | Программное обеспечение |
| РДУ | Региональное диспетчерское управление энергосистемами одного или нескольких субъектов Российской Федерации |
| РМ | Расчетная модель |
| СЭС | Солнечная электростанция |
| ТЗ | Техническое задание |
| УР | Установившийся режим |

**2. ЦЕЛИ И НАЗНАЧЕНИЕ СОЗДАНИЯ СИСТЕМЫ**

**2.1 Цели создания системы**

Целями создания Системы являются:

* Обеспечение представления СЭС в расчетных моделях структурами с эквивалентом коллекторной сети;
* Снижение трудоемкости при эквивалентировании коллекторной сети СЭС;
* Минимизация ошибок при расчете эквивалента.

**2.2 Назначение системы**

ПО «Эквивалент СЭС» предназначена для формирования эквивалентных математических моделей солнечных электростанций на основе информационной модели энергосистемы.

**2.3 Перечень диспетчерских центров АО «СО ЕЭС», в которых предполагается использовать систему**

Система разрабатывается для использования в Филиале АО «СО ЕЭС» ОДУ Сибири и РДУ операционной зоны ОДУ, содержащих в составе энергосистем солнечные электростанции.

**3. ХАРАКТЕРИСТИКА ОБЪЕКТА АВТОМАТИЗАЦИИ**

Объектом автоматизации является процедура получения эквивалентных моделей солнечных электростанций при формировании расчетных моделей установившихся режимов электроэнергетических систем ПК «RastrWin3» по данным информационной модели с применением ПО «Интеграции».

Согласно Регламенту взаимодействия филиалов АО «СО ЕЭС» ОДУ и РДУ операционной зоны ОДУ при формировании и внесении изменений в расчетные модели ОЭС, расчеты установившихся режимов и статической устойчивости осуществляются на единых расчетных моделях (ЕРМ) с использованием ПК «RastrWin3».

При этом согласно Регламенту взаимодействия структурных подразделений исполнительного аппарата и филиалов АО «СО ЕЭС» ОДУ, РДУ при актуализации данных информационной модели, в состав перечня работ, выполняемых в процессе регламентной актуализации ИМ входит формирование расчетной модели с использованием ПО «Интеграция» в ПК «RastrWin3».

Таким образом, источником данных для формирования расчетной модели является информационная модель электроэнергетической системы.

Экспорт информационной модели в расчетную, осуществляется с применением ПО «Интеграции», которая является компонентом ПК «RastrWin3».

**4. ТРЕБОВАНИЯ К СИСТЕМЕ**

Таблица 4.1 –Идентификаторы требований и их расшифровки

|  |  |
| --- | --- |
| Префикс | Тип требования |
| A | Архитектурное требование |
| С | Требование к программной или аппаратной совместимости |
| D | Требование к структуре данных |
| F | Функциональное требование |
| U | Требование к пользовательскому интерфейсу |

**4.1 Требования к архитектуре**

А01 Система должна быть представлена настольным приложением.

А02 Система должна иметь графический интерфейс.

**4.2 Требования структуре данных**

D01 В качестве входных данных должны использоваться файлы установившегося режима в формате ПК «RastrWin3» (\*.rg2).

D02 В качестве выходных данных должны использоваться файлы установившегося режима в формате ПК «RastrWin3» (\*.rg2).

**4.3 Функциональные требования**

F01 В Системе должен быть реализован ввод и загрузка файла модели энергосистемы (D01).

F02 В Системе должен быть реализован функционал:

F02.1 Выбора узлов модели, определяющих вершины будущих эквивалентов;

F02.2 Создания групп эквивалентирования для предварительно выбранных узлов модели (F02.1);

F02.3 Выбора ветвей модели для каждой группы (F02.2), подлежащих эквивалентированию.

F03 Система должна рассчитывать эквивалентные параметры элементов модели методом суммарных эквивалентных потерь:

F03.1 Выбранных ветвей (F02.3);

F03.2 Узлов, принадлежащих ветвям (F02.3);

F03.3 Генераторов, принадлежащие узлам (F03.2).

F04 На основе рассчитанных эквивалентных параметров (F03) система должна создавать объекты энергосистемы.

F05 Система должна иметь возможность добавления в модель энергосистемы, созданных эквивалентных объектов (F04).

F06 В Системе должно быть реализовано сохранения файла модели, с добавленными эквивалентными объектами (F05), на основе загруженного файла энергосистемы (F01).

**4.4 Требования к интерфейсу**

Конечное представление пользовательского интерфейса Системы должно быть проработано с Заказчиком посредством предоставления эскизов и в ходе промежуточных демонстраций Исполнителем.

U01. В графическом интерфейсе Системы должна быть возможность:

U01.1 Выбора входного файла для загрузки (F01);

U01.2 Отображения элементов загруженной модели энергосистемы;

U01.3 Подготовки элементов модели к эквивалентированию (F02);

U01.4 Отображения результатов эквивалентирования (F04);

U01.5 Внесения изменений в эквивалентные объекты (F05);

U01.6. Выбора выходного файла для сохранения.

U02. Выводимые сообщения в графическом интерфейсе должны:

U02.1 Уведомлять о результатах выполнения операций;

U02.2 Содержать описание (причины) неуспешного выполнения и пути разрешения проблемы (U02.1).

U03. Интерфейс системы должен быть основан на единой библиотеке стилей.

U04. Минимальное разрешение экрана, на котором должны отображаться элементы управления – 800 х 600.

**4.3 Требования к видам обеспечения**

**4.3.1 Требования к математическому обеспечению**

Алгоритм эквивалентирования должен быть основан на расчете эквивалентного сопротивления из условий суммарных потерь.

**4.3.3 Требования к лингвистическому обеспечению**

Основной язык интерфейса – русский (кириллица).

**4.3.4 Требования к программному обеспечению**

С01. Система должна работать под управлением операционной системы Windows 10 (X64 версии 1809 или выше);

C02. На рабочей станции должны быть установлены:

С02.1 На рабочей станции должен быть установлен  
ПК «RastrWin3» (X64 версии 2.7.1.6388 или выше);

С02.2 На рабочей станции должен быть установлен пакет .NET 6.

**4.3.4 Требования к техническому обеспечению**

C03.1 Семейство процессоров: Intel Core / Xeon или AMD FX / Zen c тактовой частотой 1 гигагерц (ГГц) или выше;

C03.2 Оперативная память: 1 гигабайт (ГБ) для 32-разрядных  
или 2 ГБ для 64‑разрядных систем;

C03.3 Место на жестком диске:16 ГБ для 32-разрядной ОС или 20 ГБ для 64‑разрядной ОС;

C03.4 Видеоадаптер: DirectX 9 или более поздняя версия;

C03.5 Экран: 800 x 600 пикселей.

**5. СОСТАВ И СОДЕРЖАНИЕ РАБОТ ПО СОЗДАНИЮ СИСТЕМЫ**

**5.1 Обязанности заказчика и исполнителя**

**5.1.1 Обязанности заказчика**

1. Рассмотрение, согласование и утверждение разработанной Исполнителем документации на Систему.
2. Предоставление необходимой информационной поддержки Исполнителю (документация и примеры) по вопросам взаимодействия Системы с другими системами, находящимися в составе ИТ‑инфраструктуры Заказчика.
3. Контроль соблюдения Исполнителем плана-графика работ, согласование корректировок.
4. Предоставление Исполнителю защищенного удаленного доступа  
   (далее - ЗУД) к своим корпоративным ресурсам и/или доступ непосредственно на объекте Заказчика.

**5.1.2 Обязанности исполнителя**

1. Разработка и предоставление на рассмотрение, согласование и утверждение Заказчику документации на Систему.
2. Выполнение работ по созданию Системы.
3. Организация промежуточных демонстраций функционально законченных частей Системы, доступа к разработанной документации на Систему с целью контроля Заказчиком правильности выбранного решения.
4. Предоставление по требованию Заказчика и в согласованные с ним сроки отчетов по выполняемым работам.
5. Использование при выполнении работ специализированных средств:

* моделирования деловых (технологических) процессов;
* хранения и контроля версий кода;
* ведения проектной документации.

1. Предоставление исходного кода Системы и дистрибутива с комплектом документов в составе;

**5.2 План-график выполнения проекта**

1. Для целей календарного планирования работ по созданию Системы Исполнитель должен разработать и предоставить на утверждение Заказчику план-график работ.
2. План-график должен содержать все элементы работ, их последовательность и сроки.
3. План-график должен актуализироваться по мере необходимости и согласовываться с Заказчиком.
4. Представленный Заказчику план-график должен быть тем же самым графиком, который используется Исполнителем для управления своими внутренними ресурсами.

Таблица 5.1 – План график выполнения работ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Этап | Содержание | Порядок приема работ | Сроки | Ответственный |
| 1. Составление технического задания | Инжиниринг требований | Утверждение технического задания | май–июнь 2022 | Разработчик, Заказчик |
| 1. Техническое проектирование | Анализ делового процесса, алгоритмизация, проработка вариантов интеграции, концептуальное проектирование | Утверждение технического проекта | сентябрь 2022 | Разработчик, Заказчик |
| 1. Разработка | Разработка программной части | Демонстрация, автономные испытания | октябрь 2022 | Разработчик |
| Разработка пользовательского интерфейса | Демонстрация, автономные испытания | ноябрь 2022 | Разработчик |
| Разработка документации | Демонстрация, автономные испытания | декабрь 2022 | Разработчик |
| 1. Автономные испытания | Проверка соответствия требованиям, наличия документации. | Протокол автономных испытаний | январь 2023 | Разработчик, Заказчик |

**6. ПОРЯДОК КОНТРОЛЯ И ПРИЕМКИ СИСТЕМЫ**

1. В рамках проекта должны быть осуществлены автономные испытания;
2. Автономные испытания должны проводиться в соответствии с программой и методикой испытаний;
3. Во время автономных испытаний должны быть проведены тесты, разработанные Исполнителем, которые используют все функции Системы, при этом должна проверяться правильность функционирования Системы;
4. В ходе испытаний должны быть выполнены следующие операции для проверки испытываемой Системы:

* Проверка того, что Система отвечает требованиям Заказчика;
* Проверка правильности получения и обработки данных;
* Проверка функционирования пользовательского интерфейса;
* Проверка документации, которая должна подтвердить, что вся документация, подлежащая поставке с Системой, имеется в наличии и отвечает требованиям.

1. Фиксация факта завершения автономных испытаний должна быть оформлена Протоколом автономных испытаний.

**7. ТРЕБОВАНИЯ К ДОКУМЕНТИРОВАНИЮ**

**7.1 Перечень подлежащих разработке комплектов и видов документов**

Исполнителем должен быть предоставлен как минимум следующий состав документации:

* Технический проект;
* Руководство пользователя;
* Программа и методика испытаний;
* Протокол автономных испытаний.

**7.2 Руководство пользователя**

1. Руководство пользователя должно содержать инструкции по эксплуатации, касающиеся функциональных возможностей и включать в себя описание функций Системы;
2. В Руководстве пользователя должны быть описаны рекомендуемые меры по устранению неисправностей, в случае их возникновения.

**7.3 Программа и методика испытаний**

1. Программа и методика испытаний должна отражать проверку всех функциональных требований к Системе, описанных в настоящем Техническом задании.
2. В методике испытаний должны описываться действия, которые должны выполняться при испытаниях Системы.
3. В описание выполняемых действий должны входить следующие позиции:

* название функции, которая должна быть протестирована;
* пошаговые описания каждого пункта испытаний;
* ссылки на проверяемые пункты настоящего Технического задания;
* формы для протоколирования результатов испытаний;
* ожидаемые результаты по каждому пункту испытаний («соответствует» / «не соответствует»).

**7.4 Формат документации**

1. Документация должна быть предоставлена на русском языке;
2. Документация на рассмотрение, согласование и утверждение должна предоставляться в электронном виде в формате «\*.docx».

**7.5 Исходный код**

Исполнитель должен предоставлять исходный текст (код) Системы любым удобным для Заказчика способом.

# Приложение Г

Таблица Г.1 – Описание класса «Node»

|  |  |  |
| --- | --- | --- |
| Название | Тип / возвращаемое значение | Описание |
| Описание класса | | |
| Node – класс для описания узлов из модели ПК «RastrWin3» | | |
| Свойства | | |
| + Index | int | Возвращает или устанавливает индекс |
| + Number | int | Возвращает или устанавливает номер |
| + Name | string | Возвращает или устанавливает наименование |
| + RatedVoltage | double | Возвращает или устанавливает номинальное напряжение |
| + DistrictNumber | int | Возвращает или устанавливает номер района |
| + TerritoryNumber | int | Возвращает или устанавливает номер территории |
| Методы | | |
| + Node(int index, int number, string name, double ratedVoltage, int districtNumber, int territoryNumber) | - | Конструктор класса Node |
| + Node(int number, string name, double ratedVoltage, int districtNumber, int territoryNumber) | - | Конструктор класса Node |

Таблица Г.2 – Описание перечисления «BranchType»

|  |  |
| --- | --- |
| Название | Описание |
| Описание класса | |
| BranchType – перечисление с типами ветвей из модели ПК «RastrWin3» | |
| Перечисление | |
| + Line | Ветви воздушных (кабельных) линий |
| + Transformer | Ветви силовых трансформаторов |

Таблица Г.3 – Описание класса «Branch»

|  |  |  |
| --- | --- | --- |
| Название | Тип / возвращаемое значение | Описание |
| Описание класса | | |
| Branch – класс для описания ветвей из модели ПК «RastrWin3» | | |
| Свойства | | |
| + Index | int | Возвращает или устанавливает индекс |
| + BranchType | BranchType | Возвращает или устанавливает тип ветви |
| + BranchStartNode | Node | Возвращает или устанавливает узел начала ветви |
| + BranchEndNode | Node | Возвращает или устанавливает узел конца ветви |
| + Name | string | Возвращает или устанавливает наименование |
| + Resistance | double | Возвращает или устанавливает активное сопротивление |
| + Inductance | double | Возвращает или устанавливает реактивное сопротивление |
| + Capacitance | double | Возвращает или устанавливает емкостную проводимость |
| + TransformationRatio | double | Возвращает или устанавливает коэффициент трансформации |
| + DistrictNumber | int | Возвращает или устанавливает номер района |
| + TerritoryNumber | int | Возвращает или устанавливает номер территории |
| + AdmissibleCurrent | double | Возвращает или устанавливает длительно допустимый ток |
| + EquipmentAdmissibleCurrent | double | Возвращает или устанавливает длительно допустимый ток оборудования |

Продолжение таблицы таблица Г.3 – Описание класса «Branch»

|  |  |  |
| --- | --- | --- |
| Название | Тип / возвращаемое значение | Описание |
| Описание класса | | |
| Branch – класс для описания ветвей из модели ПК «RastrWin3» | | |
| Методы | | |
| + Branch(int index, BranchType branchType, string name, resistance, double inductance, double capacitance, transformationRatio, int districtNumber, int territoryNumber, admissibleCurrent, double equipmentAdmissibleCurrent) | - | Конструктор класса Branch |
| + Branch(BranchType branchType, string name, resistance, double inductance, double capacitance, transformationRatio, int districtNumber, int territoryNumber, admissibleCurrent, double equipmentAdmissibleCurrent) | - | Конструктор класса Branch |

Таблица Г.4 – Описание класса «Generator»

|  |  |  |
| --- | --- | --- |
| Название | Тип / возвращаемое значение | Описание |
| Описание класса | | |
| Generator – класс для описания генераторов из модели ПК «RastrWin3» | | |
| Свойства | | |
| + Index | int | Возвращает или устанавливает индекс |
| + Number | int | Возвращает или устанавливает номер |
| + Name | string | Возвращает или устанавливает наименование |
| + GeneratorNode | Node | Возвращает или устанавливает генераторный узел |

Продолжение таблицы Г.4 – Описание класса «Generator»

|  |  |  |
| --- | --- | --- |
| Название | Тип / возвращаемое значение | Описание |
| Описание класса | | |
| Generator – класс для описания генераторов из модели ПК «RastrWin3» | | |
| Свойства | | |
| + MaxActivePower | double | Возвращает или устанавливает максимальную активную мощность |
| + PQDiagramNumber | int | Возвращает или устанавливает номер PQ диаграммы |
| Методы | | |
| + Generator(int index, int number, string name, int pQDiagramNumber, double maxActivePower) | - | Конструктор класса Generator |

Таблица Г.5 – Описание класса «ValueValidation»

|  |  |  |
| --- | --- | --- |
| Название | Тип / возвращаемое значение | Описание |
| Описание класса | | |
| ValueValidation – вспомогательный класс для валидации данных модели элементов ПК «RastrWin3» | | |
| Методы | | |
| + IsNotNaN(double value) | bool | Значение не соответствует NaN |
| + IsPositive(double value) | bool | Значение положительное число |
| + IsPositive(int value) | bool | Значение положительное число |
| + IsZeroOrPositive(double value) | bool | Значение положительное число или 0 |
| + IsZeroOrPositive(int value) | bool | Значение положительное число или 0 |
| + IsNotNullOrEmptyString(string value) | bool | Строка равна null или пустая строка |
| + IsOneOrLess | bool | Значение равно 1 или меньше |

Таблица Г.6 – Описание класса «Graph»

|  |  |  |
| --- | --- | --- |
| Название | Тип / возвращаемое значение | Описание |
| Описание класса | | |
| Graph – обобщенный класс для описания структуры графа | | |
| Свойства | | |
| + Vextexes | List<GraphVertex<T>> | Возвращает перечень вершин |
| Методы | | |
| + AddVertex(T vertexData) | void | Добавить вершину |
| + Graph() | - | Конструктор класса |
| + FindVertex(T vertexData) | GraphVertex<T> | Найти вершину по ее данным (экземпляру обобщенного типа) |
| + AddEdge(T startVertex, T endVertex, double weight) | void | Добавить ребро |

Таблица Г.7 – Описание класса «GraphVertex»

|  |  |  |
| --- | --- | --- |
| Название | Тип / возвращаемое значение | Описание |
| Описание класса | | |
| GraphVertex – обобщенный класс для описания вершины графа | | |
| Свойства | | |
| + VertexData | T | Возвращает данные по вершине (экземпляр обобщенного типа) |
| + VertexEdges | List<GraphEdge<T>> | Возвращает перечень ребер, связанных с вершиной |
| Методы | | |
| + GraphVertex(T vertexData) | void | Конструктор класса |
| + AddEdge(GraphEdge<T> edge) | void | Добавить ребро |
| + AddEdge(GraphVertex<T> vertex, double edgeWeight) | void | Добавить ребро |

Таблица Г.8 – Описание класса «GraphEdge»

|  |  |  |
| --- | --- | --- |
| Название | Тип / возвращаемое значение | Описание |
| Описание класса | | |
| GraphEdge – обобщенный класс для описания ребра графа | | |
| Свойства | | |
| + ConnectedVertex | GraphVertex<T> | Возвращает связанную вершину |
| + EdgeWeight | double | Возвращает вес ребра |
| Методы | | |
| + GraphEdge(GraphVertex<T> connectedVertex, double weight) | - | Конструктор класса |

Таблица Г.9 – Описание класса «GraphVertexInfo»

|  |  |  |
| --- | --- | --- |
| Название | Тип / возвращаемое значение | Описание |
| Описание класса | | |
| GraphVertexInfo – обобщенный класс для описания информации по вершине | | |
| Свойства | | |
| + Vertex | GraphVertex<T> | Возвращает вершину |
| + IsUnvisited | double | Возвращает или устанавливает не посещена ли вершина |
| + EdgesWeightSum | double | Возвращает или устанавливает сумму весов связанных ветвей |
| + PreviousVisitedVertex |  | Возвращает или устанавливает предыдущую посещенную вершину |
| Методы | | |
| + GraphVertexInfo(GraphVertex<T> vertex) | - | Конструктор класса |

Таблица Г.10 – Описание класса «Dijkstra»

|  |  |  |
| --- | --- | --- |
| Название | Тип / возвращаемое значение | Описание |
| Описание класса | | |
| Dijkstra – обобщенный класс для реализации алгоритма поиска кротчайшего пути методом Дейкстры | | |
| Поля | | |
| - \_graph | Graph<T> | Граф |
| - \_vertexesInfo | List<GraphVertexInfo<T>> | Перечень информации по вершинам |
| Методы | | |
| + Dijkstra(Graph<T> graph) | - | Конструктор класса |
| + AddVertexesInfo() | void | Добавить информацию по вершине |
| + GetVertexInfo(GraphVertex<T> vertex) | GraphVertexInfo<T> |  |
| + IsUnvisited | double | Возвращает или устанавливает не посещена ли вершина |
| + EdgesWeightSum | double | Возвращает или устанавливает сумму весов связанных ветвей |
| + PreviousVisitedVertex |  | Возвращает или устанавливает предыдущую посещенную вершину |

Таблица Г.11 – Описание класса «RastrConstantNames»

|  |  |  |
| --- | --- | --- |
| Название | Тип / возвращаемое значение | Описание |
| Описание класса | | |
| RastrConstantNames – статический класс строковых ключей таблиц и параметров элементов модели ПК «RastrWin3» | | |
| Поля | | |
| + NodeTable | string | Ключ таблицы «Узлы» |
| + NodeNumberColumn | string | Ключ параметра «Номер» узла |
| + NodeNameColumn | string | Ключ параметра «Наименование» узла |
| + NodeRatedVoltageColumn | string | Ключ параметра «Номинальное напряжение» узла |
| + NodeDistrictNumberColumn | string | Ключ параметра «Номер района» узла |

Продолжение таблицы Г.11 – Описание класса «RastrConstantNames»

|  |  |  |
| --- | --- | --- |
| Название | Тип / возвращаемое значение | Описание |
| Описание класса | | |
| RastrConstantNames – статический класс строковых ключей таблиц и параметров элементов модели ПК «RastrWin3» | | |
| Поля | | |
| + NodeTerritoryNumberColumn | string | Ключ параметра «Номер территории» узла |
| + BranchTable | string | Ключ таблицы «Ветви» |
| + BranchTypeColumn | string | Ключ параметра «Тип ветви» ветви |
| + BranchStartNodeColumn | string | Ключ параметра «Номер начального узла» ветви |
| + BranchEndNodeColumn | string | Ключ параметра «Номер конечного узла» ветви |
| + BranchNameColumn | string | Ключ параметра «Наименование» ветви |
| + BranchResistanceColumn | string | Ключ параметра «Активное сопротивление» ветви |
| + BranchInductanceColumn | string | Ключ параметра «Реактивное сопротивление» ветви |
| + BranchCapacitanceColumn | string | Ключ параметра «Емкостная проводимость» ветви |
| + BranchTransformationRatioColumn | string | Ключ параметра «Коэффициент трансформации» ветви |
| + BranchDistrictNumberColumn | string | Ключ параметра «Номер района» ветви |
| + BranchTerritoryNumberColumn | string | Ключ параметра «Номер территории» ветви |
| + BranchAdmissibleCurrentColumn | string | Ключ параметра «Длительно допустимый ток» ветви |
| + BranchEquipmentAdmissibleCurrentColumn | string | Ключ параметра «Длительно допустимый ток оборудования» ветви |
| + GeneratorTable | string | Ключ таблицы «Генераторы УР» |
| + GeneratorNumberColumn | string | Ключ параметра «Номер» генератора |
| + GeneratorNameColumn | string | Ключ параметра «Наименование» генератора |
| + GeneratorNodeNumberColumn | string | Ключ параметра «Номер генераторного узла» генератора |
| + GeneratorMaxActivePowerColumn | string | Ключ параметра «Максимальная активная мощность» генератора |
| + GeneratorPQDiagramNumberColumn | string | Ключ параметра «Номер PQ диаграммы» генератора |

Таблица Г.12 – Описание класса «RastrProvider»

|  |  |  |
| --- | --- | --- |
| Название | Тип / возвращаемое значение | Описание |
| Описание класса | | |
| RastrProvider – статический класс синхронных методов взаимодействия с моделью «RastrWin3» | | |
| Поля | | |
| - \_rastr | Rastr | Экземпляр класса Rastr |
| Методы | | |
| + LoadFileByTemplate(string loadFilePath, string templateFilePath) | void | Загрузить файл модели ПК «RastrWin3» по шаблону |
| + SaveFileByTemplate(string saveFilePath, string templateFilePath) | void | Сохранить файл модели ПК «RastrWin3» по шаблону |
| - CheckFileExistence(string filePath) | void | Проверить наличия файла по указанному пути |
| - CheckFileFormat(string filePath, string fileFormat) | void | Проверить соответствие файла заданному формату |
| - GetElementParameterValue<T>(ITable table, string columnName, int elementIndex) | void | Получить значение параметра элемента |
| - SetElementParameterValue<T>(ITable table, string columnName, int elementIndex, T parameterValue) | void | Установить значение параметра элемента |
| - GetNodeByIndex(int nodeIndex, ITable nodesTable) | Node | Получить узел модели по индексу |
| + GetNodes() | List<Node> | Получить узлы модели |
| + GetNodesCount() | int | Получить количество узлов в модели |
| - AddNode(Node node, ITable nodeTable) | void | Добавить узел в модель |
| + AddNodes(List<Node> nodes) | void | Добавить узлы в модель |
| + DeleteNodes(List<Node> nodes) | void | Удалить узлы из модели |
| - GetBranchByIndex(int branchIndex, ITable branchesTable) | Branch | Получить ветвь модели по индексу |
| + GetBranches(List<Node> nodes) | List<Branch> | Получить ветви модели |
| + GetBranchesCount() | int | Получить количество ветвей в модели |
| - AddBranch(Branch branch, ITable branchTable) | void | Добавить ветвь в модель |
| + AddBranches(List<Branch> branches) | void | Добавить ветви в модель |

Продолжение таблицы Г.12 – Описание класса «RastrProvider»

|  |  |  |
| --- | --- | --- |
| Название | Тип / возвращаемое значение | Описание |
| Описание класса | | |
| RastrProvider – статический класс синхронных методов взаимодействия с моделью «RastrWin3» | | |
| Поля | | |
| - \_rastr | Rastr | Экземпляр класса Rastr |
| Методы | | |
| + DeleteBranches(List<Branch> branches) | void | Удалить ветви из модели |
| + DeleteBlankBranches() | void | Удалить ветви с отсутствующим узлом начала или конца |
| - GetGeneratorByIndex(int generatorIndex, ITable generatorsTable) | Generator | Получить генератор модели по индексу |
| + GetGenerators(List<Node> nodes) | List<Generator> | Получить генераторы модели |
| + GetGeneratorsCount() | int | Получить количество генераторов в модели |
| + UpdateGeneratorsNodes(Node newGeneratorsNode, List<Generator> generators) | void | Обновить генераторные узлы генераторов |

Таблица Г.13 – Описание класса «RastrProviderAsync»

|  |  |  |
| --- | --- | --- |
| Название | Тип / возвращаемое значение | Описание |
| Описание класса | | |
| RastrProviderAsync – статический класс асинхронных методов взаимодействия с моделью «RastrWin3» | | |
| Методы | | |
| + GetNodesAsync(CancellationToken token) | Task<List<Node>> | Получить узлы модели |
| + GetGeneratorsAsync(List<Node> nodes, CancellationToken token) | Task<List<Generator>> | Получить генераторы модели |
| + GetBranchesAsync(List<Node> nodes, CancellationToken token) | Task<List<Branch>> | Получить ветви модели |

Таблица Г.14 – Описание класса «MainWindowViewModel»

|  |  |  |
| --- | --- | --- |
| Название | Тип / возвращаемое значение | Описание |
| Описание класса | | |
| MainWindowViewModel – класс модели представления основной формы приложения | | |
| Свойства | | |
| + Nodes | ObservableCollection<Node> | Возвращает или устанавливает перечень загруженных узлов |
| + Branches | ObservableCollection<Branch> | Возвращает или устанавливает перечень загруженных ветвей |
| + Generators | List<Generator> | Возвращает или устанавливает перечень загруженных генераторов |
| + EquivalenceNodes | ObservableCollection <EquivalenceNodeViewModel> | Возвращает или устанавливает перечень узлов (вершин эквивалентов) |
| + SelectedNode | Node | Возвращает или устанавливает выбранный в представлении узел |
| + SelectedEquivalenceNode | EquivalenceNodeViewModel | Возвращает или устанавливает выбранный в представлении узел (вершину эквивалента) |
| + SelectedEquivalenceGroup | EquivalenceGroupViewModel | Возвращает или устанавливает выбранный в представлении группу эквивалентирования |
| + SelectedBranch | Branch | Возвращает или устанавливает выбранную в представлении ветвь |
| + SelectedEquivalenceBranch | Branch | Возвращает или устанавливает выбранную в представлении ветвь для эквивалентирования |
| + IsFileDownloading | bool | Возвращает или устанавливает состояние процесса загрузки файла |
| + IsFileDownloaded | bool | Возвращает или устанавливает состояние результата загрузки файла |
| + IsModelChanged | bool | Возвращает или устанавливает состояние изменения модели для эквивалентирования |

Продолжение таблицы Г.14 – Описание класса «MainWindowViewModel»

|  |  |  |
| --- | --- | --- |
| Название | Тип / возвращаемое значение | Описание |
| Описание класса | | |
| MainWindowViewModel – класс модели представления основной формы приложения | | |
| Свойства | | |
| + ValidateErrors | ObservableCollection <Exception> | Возвращает или устанавливает перечень ошибок в ходе проверки модели перед эквивалентированием |
| + IsCalculatedEquivalent | bool | Возвращает или устанавливает состояние результата расчета эквивалента |
| + MaxStatusBarValue | double | Возвращает или устанавливает максимальное значение строки состояния (progress bar) |
| + CurrentStatusBarValue | double | Возвращает или устанавливает текущие значение строки состояния (progress bar) |
| + Token | CancellationToken | Возвращает или устанавливает токен прерывания операции (задачи) |
| Методы | | |
| + DownloadFile() | void | Загрузить файл модели |
| + CancelDownloadingFile() | void | Отменить загрузку файла модели |
| - CanCancelDownloadingFile() | bool | Возможность отмены загрузки файла |
| + SaveFile() | void | Сохранить модель с эквивалентами в файл |
| - CanSaveFile() | bool | Возможность сохранения модели с эквивалентом в файл |
| + CloseApplication() | void | Закрыть приложение |
| + AddNodeToEquivalenceNodes() | void | Добавить узел модели в перечень узлов (вершин эквивалентов) |
| + RemoveNodeFromEquivalenceNodes() | void | Удалить узел (вершину эквивалента) из перечня |
| + AddEquivalenceGroupToEquivalenceNode() | void | Добавить группу эквивалентирования |
| + DeleteEquivalenceGroupFromEquivalenceNode() | void | Удалить группу эквивалентирования |
| + AddBranchToEquivalenceGroup() | void | Добавить ветвь модели в перечень ветвей для эквивалентирования |

Продолжение таблицы Г.14 – Описание класса «MainWindowViewModel»

|  |  |  |
| --- | --- | --- |
| Название | Тип / возвращаемое значение | Описание |
| Описание класса | | |
| MainWindowViewModel – класс модели представления основной формы приложения | | |
| Методы | | |
| + RemoveNodeFromEquivalenceNodes() | void | Удалить узел (вершину эквивалента) из перечня |
| + AddEquivalenceGroupToEquivalenceNode() | void | Добавить группу эквивалентирования |
| + DeleteEquivalenceGroupFromEquivalenceNode() | void | Удалить группу эквивалентирования |
| + AddBranchToEquivalenceGroup() | void | Добавить ветвь модели в перечень ветвей для эквивалентирования |
| + RemoveBranchFromEquivalenceGroup() | void | Удалить ветвь для эквивалентирования из перечня |
| + ValidateModel() | void | Выполнить проверку модели для эквивалентирования |
| - CanValidateModel() | bool | Возможность проверки модели для эквивалентирования |
| + CalculateEquivalent() | void | Рассчитать эквивалент |
| - CanCalculateEquivalent() | bool | Возможность рассчитать эквивалент |
| + MainWindowViewModel() | - | Конструктор класса |

Таблица Г.15 – Описание класса «EquivalenceNodeViewModel»

|  |  |  |
| --- | --- | --- |
| Название | Тип / возвращаемое значение | Описание |
| Описание класса | | |
| EquivalenceNodeViewModel – класс модели представления узла (вершины эквивалента) | | |
| Свойства | | |
| + NodeElement | Node | Возвращает или устанавливает узел модели |
| + EquivalenceGroups | EquivalenceGroupViewModel | Возвращает или устанавливает группы эквивалентирования |
| Методы | | |
| + EquivalenceNodeViewModel(Node node) | - | Конструктор класса |

Таблица Г.16 – Описание класса «EquivalenceGroupViewModel»

|  |  |  |
| --- | --- | --- |
| Название | Тип / возвращаемое значение | Описание |
| Описание класса | | |
| EquivalenceGroupViewModel – класс модели представления группы эквивалентирования | | |
| Свойства | | |
| + Id | int | Возвращает или устанавливает индексы группы |
| + Name | string | Возвращает или устанавливает наименование группы |
| + EquivalenceBranches | ObservableCollection<Branch> | Возвращает или устанавливает перечень ветвей для эквивалентирования |
| + EquivalenceNodes | List<Node> | Возвращает или устанавливает узлы группы эквивалентирования |
| + EquivalenceGenerators | List<Generator> | Возвращает или устанавливает генераторы группы эквивалентирования |
| + IntermediateEquivalentNode | Node | Возвращает или устанавливает промежуточный узел эквивалента |
| + GeneratorEquivalentNode | Node | Возвращает или устанавливает генераторный узел эквивалента |
| + EquivalentBranches | ObservableCollection<Branch> | Возвращает или устанавливает ветви эквивалента |
| Методы | | |
| + EquivalenceGroupViewModel(int id, string name) | - | Конструктор класса |

Таблица Г.17 – Описание класса «Equivalentator»

|  |  |  |
| --- | --- | --- |
| Название | Тип / возвращаемое значение | Описание |
| Описание класса | | |
| Equivalentator – статический класс с методами проверки модели и расчета эквивалента | | |
| Методы | | |
| + GetNodesOfEquivalenceGroup(EquivalenceGroupViewModel equivalenceGroup) | List<Node> | Получить узлы группы эквивалентирования |
| + GetGeneratorsOfEquivalenceGroup(List<Node> nodesOfEquivalenceGroup, List<Generator> generators) | List<Generator> | Получить генераторы группы эквивалентирования |
| + GetGraphOfEquivalenceGroup(EquivalenceGroupViewModel equivalenceGroup, List<Node> nodesOfEquivalenceGroup) | Graph<Node> | Получить граф группы эквивалентирования |
| + GetEquivalenceBranchToGeneratorsPower(EquivalenceNodeViewModel equivalenceNode, EquivalenceGroupViewModel equivalenceGroup, List<Generator> generatorsOfEquivalenceGroup, Dijkstra<Node> dijkstra) | Dictionary<Branch, double> | Получить словарь ветвей с сопоставленной им мощностью генераторов |
| - FindBranchInEquivalenceGroup(EquivalenceGroupViewModel equivalenceGroup, Node firstNode, Node secondNode) | Branch | Найти ветвь в группе эквивалентирования |
| - GetTotalGeneratorsPower(List<Generator> generatorsOfEquivalenceGroup) | double | Получить суммарную мощность генераторов группы эквивалентирования |
| + GetEquivalentBranches(EquivalenceNodeViewModel equivalenceNode, Dictionary<Branch, double> equivalenceBranchToGeneratorsPower, List<Generator> generatorsOfEquivalenceGroup, EquivalenceGroupViewModel equivalenceGroup) | void | Получить эквивалент ветвей группы эквивалентирования |
| + GetIntermediateEquivalentNode(EquivalenceNodeViewModel equivalenceNode, EquivalenceGroupViewModel equivalenceGroup) | void | Получить промежуточный узел эквивалента |
| + GetGeneratorEquivalentNode(EquivalenceGroupViewModel equivalenceGroup) | void | Получить генераторный узле эквивалента |

Продолжение таблицы Г.17 – Описание класса «Equivalentator»

|  |  |  |
| --- | --- | --- |
| Название | Тип / возвращаемое значение | Описание |
| Описание класса | | |
| Equivalentator – статический класс с методами проверки модели и расчета эквивалента | | |
| Методы | | |
| + SetEquivalentNodeToEquivalentBranch(EquivalenceNodeViewModel equivalenceNode, EquivalenceGroupViewModel equivalenceGroup) | void | Установить эквивалентным ветвям эквивалентные узлы |
| + IsHasEquivalenceBranchesDuplicates(EquivalenceGroupViewModel equivalenceGroup) | bool | Имеются ли дубликаты ветвей в группе эквивалентирования |
| + IsOneGeneratorsRatedVoltageLevel(List<Generator> generatorsOfEquivalenceGroup) | bool | Генераторы эквивалентной группы с единым генераторным напряжением |

Таблица Г.18 – Описание класса «BranchTypeConverter»

|  |  |  |
| --- | --- | --- |
| Название | Тип / возвращаемое значение | Описание |
| Описание класса | | |
| BranchTypeConverter – класс прямой и обратной конвертации типа BranchType для отображения в View | | |
| Методы | | |
| - Convert(object value, Type targetType, object parameter, CultureInfo culture) | object | Преобразовать BranchType в string |
| - ConvertBack(object value, Type targetType, object parameter, CultureInfo culture) | object | Преобразовать string в BranchType |