**Содержание**

[Перечень сокращений, условных обозначений, терминов 7](#_Toc437553278)

[Введение 8](#_Toc437553279)

[1 Техническое задание на создание системы 9](#_Toc437553280)

[1.1 Назначение и цели создания системы 9](#_Toc437553281)

[1.2 Характеристика объекта автоматизации 9](#_Toc437553282)

[1.2.1 Общее описание 9](#_Toc437553283)

[1.2.2 Структура и принципы функционирования 9](#_Toc437553284)

[1.3 Общие требования к системе 10](#_Toc437553285)

[1.4 Требования к функциям, выполняемым системой 10](#_Toc437553286)

[1.4.1 Функция загрузки обрабатываемых данных 10](#_Toc437553287)

[1.4.2 Функция морфологического анализа текста 10](#_Toc437553288)

[1.4.3 Функция статистического анализа слов и словосочетаний 11](#_Toc437553289)

[1.4.4 Функция кластерного анализа текста и определения набора терминов 11](#_Toc437553290)

[2 Информационное обеспечение системы 12](#_Toc437553291)

[2.1 Организация входных данных 12](#_Toc437553292)

[2.2 Организация обработки данных 12](#_Toc437553293)

[2.3 Организация выдачи информации 12](#_Toc437553294)

[3 Алгоритмическое обеспечение системы 13](#_Toc437553295)

[3.1 Алгоритм привязки данных 13](#_Toc437553296)

[4 Программное обеспечение системы 14](#_Toc437553297)

[4.1 Выбор компонентов программного обеспечения 14](#_Toc437553298)

[4.1.1 Инструментальные средства разработки и язык программирования 14](#_Toc437553299)

[4.1.2 Вспомогательное программное обеспечение 15](#_Toc437553300)

[4.2 Разработка прикладного программного обеспечения 16](#_Toc437553301)

[4.2.1 Структура прикладного программного обеспечения 16](#_Toc437553302)

[4.2.2 Модуль обмена данными 18](#_Toc437553303)

[4.2.3 Модуль привязки данных 21](#_Toc437553304)

[4.2.4 Модуль сбора данных 22](#_Toc437553305)

[4.2.5 Модуль формирования запросов 23](#_Toc437553306)

[4.2.6 Модуль формирования файлового хранилища 24](#_Toc437553307)

[4.2.7 Модуль обеспечения бесперебойной передачи данных 24](#_Toc437553308)

[4.2.8 Вспомогательный модуль 25](#_Toc437553309)

[4.2.9 Модуль перехвата сообщений 27](#_Toc437553310)

[4.2.10 Модуль конфигурации 28](#_Toc437553311)

[4.2.11 Модуль панели управления интеграцией 29](#_Toc437553312)

[4.3 Особенности эксплуатации и сопровождения системы 29](#_Toc437553313)

[4.4 Интерфейс пользователя с системой 31](#_Toc437553314)

[4.4.1 Руководство пользователя 31](#_Toc437553315)

[4.4.2 Инсталляция и настройка 38](#_Toc437553316)

[4.4.3 Исключительные ситуации и их обработка 38](#_Toc437553317)

[Заключение 39](#_Toc437553318)

[Список литературы 40](#_Toc437553319)

[Приложение А 42](#_Toc437553320)

# Перечень сокращений, условных обозначений, терминов

ЖКХ – жилищно-коммунальное хозяйство;

# Введение

1. Техническое задание на создание системы

В данном разделе приводится техническое задание на разработку системы статистического, морфологического, кластерного и онтологического анализа данных для ЖКХ.

* 1. Назначение и цели создания системы

Данная система предназначена для общего круга лиц и предусматривает использования ее любым пользователем, которому требуется провести комплексный анализ текстов.

Система создается для того, чтобы:

1. автоматизировать расчет статистических характеристик анализируемых текстов;
2. автоматизировать определение списка терминов текста и степень их значимости.
   1. Характеристика объекта автоматизации

Объектом автоматизации системы является автоматическое выделение терминов из текста по предметной области.

* + 1. Общее описание

Данная система предназначается для автоматизации проведения анализа текстовых данных. Система должна включать в себя статистический, морфологический, кластерный анализы для определения набора терминов обрабатываемого текста. Статистический анализ теста должен проводиться как для взятых слов, так и для словосочетаний. Морфологический анализ должен быть организован с применением специального программного обеспечения “Mystem”. Кластерный анализ должен быть реализован на основе метода К-средних.

* + 1. Структура и принципы функционирования

Структура разрабатываемой системы должна быть последовательной и должна состоять из следующих ключевых этапов:

1. выбор обрабатываемых файлов и последующая их загрузка в память;
2. проведение морфологического анализа загруженных текстов;
3. расчет статистические характеристики для отдельных слов и словосочетаний;
4. поиск терминов.
   1. Общие требования к системе

В разделе приводится описание основных требований на разработку системы статистического, морфологического, кластерного и онтологического анализа данных для ЖКХ.

* 1. Требования к функциям, выполняемым системой

В разделе приводиться описание требований к основным функциям, выполняемым системой статистического, морфологического, кластерного и онтологического анализа данных для ЖКХ.

* + 1. Функция загрузки обрабатываемых данных

Загрузка данных для обработки в разрабатываемом программном обеспечении должна производиться из текстовых файлов. Данные этих файлов должны быть загружены полностью в память программы. В программном обеспечении должна быть реализована возможность как загрузки одного файла, так и параллельное чтение данных из файлов в нескольких потоках.

* + 1. Функция морфологического анализа текста

Функция должна выделять слова из загруженных на обработку текстов. Исключать знаки препинания, пробелы и другие разделители. Функция должна приводить все слова в каноническую форму для повышения точности выполнения последующих функций программы. Для реализации данной функции является обязательным использование специального программного обеспечения “Mystem”. Приложение должно быть встроено в реализуемое программное обеспечение, а обмен данными построен на основе файлового взаимодействия.

* + 1. Функция статистического анализа слов и словосочетаний

Назначение функции зкалючается в произведении расчетов статистических характеристик слов и словосочетаний. Этими характеристиками являются:

1. частота встречаемости слова/словосочетания в тексте;
2. статистическая значимость встречаемости слов словосочетания в тексте по методу Mutual Information;
3. степень взаимосвязи двух слов по методу T-Score;
4. степень значимости словосочетания в тексте по методу Log-Likelihood;
5. степень значимости словосочетания в тексте по методу TF×IDF.
   * 1. Функция кластерного анализа текста и определения набора терминов

Функция осуществляет поиск слов/словосочетаний-терминов в тексте на основе данных, полученных в ходе статистического анализа.

1. Информационное обеспечение системы

В данном разделе рассматриваются структуры источников данных для системы статистического, морфологического, кластерного и онтологического анализа данных для ЖКХ. Отдельно приводится описание онтологии предметной области.

* 1. Организация входных данных

Основными входными данными системы являются текстовые файлы формата .pdf, .txt. Файлы могут иметь произвольную структуру.

* 1. Организация обработки данных

Обработка данных выполняется по порядку в соответствии с описанием функций программы в разделе техническое задание. Каждая функция выполняется только при успешном выполнении всех предыдущих. Перед вызовом каждой функции возможна предварительная настройка их параметров.

* 1. Организация выдачи информации

Выдача промежуточных и конечного результатов работы системы осуществляется с помощью специальных элементов пользовательского интерфейса. Также результаты, полученные в ходе вычислений, могут быть экспортированы в файл формата .json для последующего использования, например при визуализации данных.

* 1. Онтология предметной области

Онтология дает возможность представить данные в виде упорядоченной иерархической структуры. В рамках разработанной системы для анализа слабоструктурированных данных, текстов, была создана онтология, которая отражает особенности предметной области.

На рисунке # представлена онтология, дающая общее представление об основных объектах сферы ЖКХ.

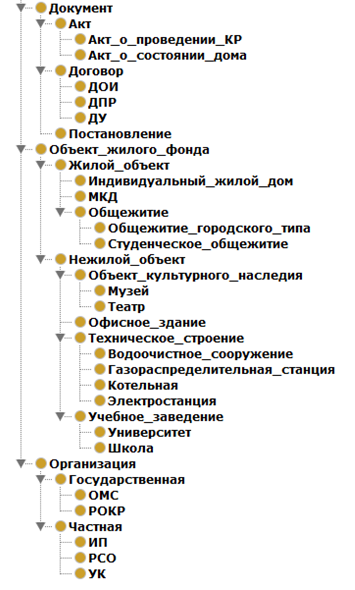


Рисунок #. Структура онтологии

Онтология подразделяется на 3 группы классов:

1. объекты жилого фонда, многоквартирные дома, муниципальные учреждения и т.д.;
2. организации, производящие контроль и управление над объектами жилого фонда;
3. документы, которые описывают основные положения бизнес-процессов управления и контроля.

В таблице # приводится подробное описание представленной онтологии.

Таблица #. Описание онтологии предметной области

| Название | Описание | Data property | Примечание |
| --- | --- | --- | --- |
| Документ | Класс «Документ» объединяет в себе все возможные документы касающихся состояния дома и управления им. | Дата начала действия: dateTime,  Дата окончания действия: dateTime,  Дата подписания: dateTime,  Номер: string | Содержит в себе подклассы: Акт и Договор. |
| Акт | Класс «Акт» предназначен для сбора информации об актах на состояние дома и проведения КР. | Положения: string,  Состояние дома: string | Является подклассом класса «Документ».  Содержит в себе подклассы: Акт о проведении КР и Акт о состоянии дома. |
| Акт о проведении  Капитального ремонта (КР) | Класс «Акт о проведении КР» предназначен для сбора информации о проведении в домах КР. | Дата проведения КР: dateTime | Является подклассом класса «Акт». |
| Акт о состоянии дома | Класс «Акт о состоянии дома» предназначен для сбора информации о состоянии дома. | - | Является подклассом класса «Акт». |
| Договор | Класс «Договор» предназначен для сбора информации о договорах на управление, поставку ресурсов и общего имущества. | Обязательства: string | Является подклассом класса «Документ».  Содержит в себе подклассы: Договор общего имущества, Договор поставки ресурсов и Договор управления. |
| Договор общего имущества (ДОИ) | Класс «Договор общего имущества» предназначен для сбора информации о договорах общего имущества. | Организация: string | Является подклассом класса «Договор». |
| Договор поставки ресурсов (ДПР) | Класс «Договор поставки ресурсов» предназначен для сбора информации о поставке ресурсов. | РСО: string | Является подклассом класса «Договор». |
| Договор управления (ДУ) | Класс «Договор управления» предназначен для сбора информации о договорах управления. | Управляющая организация: string | Является подклассом класса «Договор». |
| Постановление | Класс «Постановление» предназначен для сбора информации о постановлениях. | Закон: string | Является подклассом класса «Документ». |
| Объект жилого фонда | Класс «Объект жилого фонда» предназначен для сбора информации о жилых и нежилых объектах. | Аварийность: boolean,  Адрес: string,  Год постройки: dateTime,  Описание: string,  Площадь: decimal,  Этажность: int | Содержит в себе подклассы: Жилой объект и Нежилой объект |
| Жилой объект | Класс «Жилой объект» предназначен для сбора информации о индивидуальных жилых домах, многоквартирных домах и общежитие. | Жилая площадь: decimal,  Тип управления: string,  Число жителей: int | Является подклассом класса «Объект жилого фонда».  Содержит в себе подклассы: Индивидуальный жилой дом, Многоквартир-ный дом и Общежитие |
| Индивидуаль-ный жилой дом | Класс «Индивидуаль-ный жилой дом» предназначен для сбора информации о индивидуальных жилых домах. | Наличие канализации: boolean,  Площадь участка: decimal | Является подклассом класса «Жилой объект». |
| Многоквартир-ный дом (МКД) | Класс «Многоквар-тирный дом» предназначен для сбора информации о многоквартирных домах | Количество квартир: int,  Количество лифтов: int,  Количество подъездов: int,  Наличие технического этажа: boolean | Является подклассом класса «Жилой объект». |
| Общежитие | Класс «Общежитие» предназначен для сбора информации о общежитиях городского типа и студенческих общежитиях | Количество комнат: int,  Наличие вахты: boolean | Является подклассом класса «Жилой объект». Содержит в себе подклассы: Общежитие городского типа и Студенческое общежитие |
| Общежитие городского типа | Класс «Общежитие городского типа» предназначен для сбора информации о общежитиях городского типа. | - | Является подклассом класса «Общежитие». |
| Студенческое общежитие | Класс «Студенческое общежитие» предназначен для сбора информации о студенческих общежитиях. | - | Является подклассом класса «Общежитие». |
| Нежилой объект | Класс «Нежилой объект» предназначен для сбора информации о объектах культурного наследия, офисных зданиях, технических строениях и учебных заведениях. | - | Является подклассом класса «Объект жилого фонда».  Содержит в себе подклассы: Объект культурного наследия, Офисное здание, Техническое строение и Учебное заведение |
| Объект культурного наследия | Класс «Объект культурного наследия» предназначен для сбора информации о музеях и театрах. | Максимальное число посетителей: int | Является подклассом класса «Нежилой объект».  Содержит в себе подклассы:  Музей и Театр |
| Музей | Класс «Музей» предназначен для сбора информации о музеях. | Количество выставочных помещений: int | Является подклассом класса «Объект культурного наследия». |
| Театр | Класс «Театр» предназначен для сбора информации о театрах. | Количество концертных залов: int | Является подклассом класса «Объект культурного наследия». |
| Офисное здание | Класс «Офисное здание» предназначен для сбора информации о офисных зданиях. | Количество организаций: int | Является подклассом класса «Объект жилого фонда». |
| Техническое строение | Класс «Техническое строение» предназначен для сбора информации о водоочистных сооружениях, газораспределитель-ных станциях, котельных и электростанциях. | Вид ресурса: string,  Вид услуги: string,  Количество потребителей: int,  Производственная мощность: decimal | Является подклассом класса «Нежилой объект».  Содержит в себе подклассы:  Водоочистное сооружение, Газораспредели-тельная станция, Котельная и Электростанция |
| Водоочистное сооружение | Класс «Водоочистное сооружение» предназначен для сбора информации о водоочистных сооружениях. | Тип фильтрации: string | Является подклассом класса «Техническое строение». |
| Газораспреде-лительная станция | Класс «Газораспреде-лительная станция» предназначен для сбора информации о газораспределитель-ных станциях. | Диаметр труб: decimal,  Объем распределяемого газа: decimal | Является подклассом класса «Техническое строение». |
| Котельная | Класс «Котельная» предназначен для сбора информации о котельных. | Количество котлов: int,  Марка котла: string | Является подклассом класса «Техническое строение». |
| Электро-станция | Класс «Электро-станция» предназначен для сбора информации о электростанциях. | Вид подстанции: string,  Напряжение: decimal | Является подклассом класса «Техническое строение». |
| Учебное заведение | Класс «Учебное заведение» предназначен для сбора информации о университетах и школах. | - | Является подклассом класса «Нежилой объект».  Содержит в себе подклассы:  Университет и Школа |
| Университет | Класс «Университет» предназначен для сбора информации о университетах. | - | Является подклассом класса «Учебное заведение». |
| Школа | Класс «Школа» предназначен для сбора информации о школах. | - | Является подклассом класса «Учебное заведение». |
| Организация | Класс «Организация» предназначен для сбора информации о государственных и частных организациях. | ИНН: string, Количество инженеров: int,  Количество работников: int,  Количество рабочих: int,  Контактные данные: string,  КПП: string,  Название: string,  ОГРН: string | Содержит в себе подклассы: Государственная и Частная. |
| Государствен-ная | Класс «Государственная» предназначен для сбора информации о органах местного самоуправления и региональных операторах капитального ремонта. | Официальный представитель: string | Является подклассом класса «Организация».  Содержит в себе подклассы: Органы местного самоуправления и Региональный оператор капитального ремонта. |
| Органы местного самоуправ-ления (ОМС) | Класс «Органы местного самоуправ-ления» предназначен для сбора информации о органах местного самоуправления. | - | Является подклассом класса «Государственная». |
| Региональный оператор капитального ремонта (РОКР) | Класс «Региональный оператор капитального ремонта» предназначен для сбора информации о региональных операторах капитального ремонта. | - | Является подклассом класса «Государственная». |
| Частная | Класс «Частная» предназначен для сбора информации об индивидуальных предпринимателях, ресурсоснабжающих организациях и управляющих компаниях. | Директор: string,  Одобренный подрядчик: boolean | Является подклассом класса «Организация».  Содержит в себе подклассы: Индивидуальный предпри-ниматель, Ресурсо-снабжающая организация и Управляющая компания. |
| Индивидуаль-ный предпри-ниматель (ИП) | Класс «Индивидуаль-ный предприниматель» предназначен для сбора информации об индивидуальных предпринимателях. | - | Является подклассом класса «Частная». |
| Ресурсо-снабжающая организация (РСО) | Класс «Ресурсо-снабжающая организация» предназначен для сбора информации о ресурсоснабжающих организациях. | Тип обслуживания: string | Является подклассом класса «Частная». |
| Управляющая компания (УК) | Класс «Управляющая компания» предназначен для сбора информации об управляющих компаниях. | - | Является подклассом класса «Частная». |

Классы онтологии взаимосвязаны с помощью объектных слов, перечень которых представлен на рисунке #.

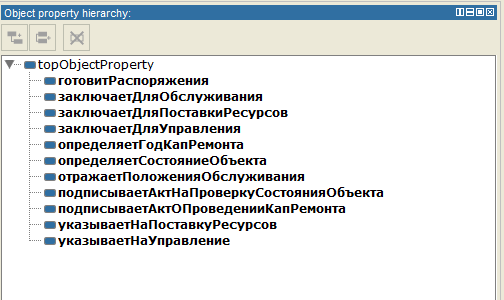


Рисунок #. Объектные свойства онтологии

В таблице # представлено подробное описание объектных свойств онтологии.

Таблица #. Описание объекных свойств онтологии

| Наименование | Описание |
| --- | --- |
| Готовит распоряжения | Свойство относится к государственной организации, а именно органам местного самоуправления. Указывает на подготовку ОМС распоряжения |
| Заключает для обслуживания | Свойство относится к частной организации, а именно индивидуальным предпринимателям. Указывает на подготовку договора общего имущества, заключенного для обслуживания объекта |
| Заключает для поставки ресурсов | Свойство относится к частной организации, а именно ресурсоснабжающим организациям. Указывает на заключение договора поставки энергетических ресурсов до объекта |
| Заключает для управления | Свойство относится к частной организации, а именно управляющим компаниям. Указывает на заключение договора управления объектом жилого фонда |
| Определяет год капитального ремонта | Свойство относится к документам, а именно актам о проведении капитального ремонта. Указывает на отношение договора к объекту жилого фонда |
| Определяет состояние объекта | Свойство относится к документам, а именно актам о состоянии дома. Указывает на отношение договора к объекту жилого фонда |
| Отражает положения обслуживания | Свойство относится к документам, а именно договорам общего имущества. Указывает на отношение договора к жилому объекту |
| Подписывает акт на поверку состояния объекта | Свойство относится к государственной организации, а именно региональному оператору капитального ремонта. Указывает на подписание акта поверки состояния объекта РОКР |
| Подписывает акт о проведении капитального ремонта | Свойство относится к государственной организации, а именно региональному оператору капитального ремонта. Указывает на подписание акта о проведении капитального ремонта РОКР |
| Указывает на поставку ресурсов | Свойство относится к договору на поставку ресурсов, а именно поставку ресурсов ресурсоснабжающими организациями. Указывает на заключение договора поставки энергетических ресурсов до объектов жилого фонда |
| Указывает на управление | Свойство относится к договору управления, а именно управления объектами жилого фонда. Указывает на заключение договора управления |

1. Алгоритмическое обеспечение системы

В разделе рассматриваются алгоритмы, разработанные в процессе создания системы статистического, морфологического, кластерного и онтологического анализа данных для ЖКХ. Реализация алгоритмов на ЯВУ C# представлена в приложении А.

* 1. Алгоритм кластеризации

FCM (Fuzzy Classifier Means) - это алгоритм кластеризации, с помощью которого производится автоматическая классификация множества объектов, которые задаются векторами признаков в пространстве признаков. В случае рассматриваемой системы объектами является множество слов и словосочетаний. Их векторы признаков – наборы характеристик, получаемые в результате статистического анализа.

На схеме # представлена схема подпрограммы, которая реализует алгоритм кластеризации.

Схема #. Блок-схема подпрограммы FCM-алгоритма

Входными данными для алгоритма являются:

1. количество итераций, определяющих точность выполнения алгоритма;
2. центры кластеров;
3. матрица, описывающая степень принадлежности каждого из объектов к центрам кластеров;
4. параметр нечеткости кластеризации. Рекомендуемое значение параметра выбирается в пределах ~1,5...2. При реализации алгоритма величина параметра была установлена в значение 1,6;
5. параметр сходимости алгоритма (уровень точности). Значение параметра установлено в 0.001.

На первом шаге алгоритма производится заполнение матрицы принадлежности объектов установленным центрам кластеров. Значения матрицы устанавливаются случайным образом, но с условием, что сумма значений вектора принадлежности отдельного объекта не превышает 1.0.

На втором шаге производится запуск основного цикла алгоритма кластеризации данных, регулируются значения центров кластеров по формуле:

На третьем этапе происходит корректировка значений матрицы принадлежности по формуле:

По завершении каждой итерации цикла алгоритма производятся проверки:

1. сравнения числа уже выполненных итераций с заранее предустановленным количеством;
2. сравнения параметра сходимости алгоритма и матричной нормы, которая должна быть больше для продолжения работы алгоритма.

При выполнении хотя бы одного из условий алгоритм считается завершенным.

Результом работы алгоритма является итоговая матрица принадлжености объектов к кадому из кластеров.

1. Программное обеспечение системы

В разделе приводится описание инструментальных средств разработки, применяемых в процессе создания подсистемы интеграции, основных модулей приложения, руководства пользования подсистемой и процесса ее установки и настройки.

* 1. Выбор компонентов программного обеспечения

В подразделе рассматриваются основные и вспомогательные инструментальные средства разработки, которые были использованы в процессе создания приложения интеграции.

* + 1. Инструментальные средства разработки и язык программирования

Подсистема интеграции для региональной системы объектового учета «АИС: Объектовый учет» и федерального портала «Реформа ЖКХ» была разработана с применением технологий платформы .NET.

Серверная часть подсистемы, обеспечивающая сбор и обмен данными с Реформой, реализована в виде консольного приложения, хотя в требованиях упоминается о создании службы интеграции. Смена подхода объясняется тем, что в процессе разработки возникла необходимость полноценной отладки создаваемого программного обеспечения. Технология служб .NET не поддерживает эту особенность, поэтому выбор был сделан в пользу консольного приложения. Интерфейс панели управления подсистемой интеграции реализован с применением технологии ASP.NET MVC 4.

В ходе разработки подсистемы интеграции были использованы следующие языки программирования:

1. C# 4.0., с помощью которого были созданы подсистема обмена данными и серверная часть панели управления интеграции для пользователя управляющей организации;
2. язык гипертекстовой разметки HTML 5 для создания web-страниц;
3. язык описания внешнего вида web-страниц CSS 3;
4. встраиваемый язык программирования JavaScript для реализации интерактивных сценариев на web-страницах;
5. язык доступа к данным Transact-SQL для организации процессов управления данными на стороне СУБД.

В качестве основной IDE использовалась Visual Studio 13 версии. Программный пакет используется в компании «АИС: Город» при разработке программных решений, так как фирма является сертифицированным партнером Microsoft.

* + 1. Вспомогательное программное обеспечение

Для управления проектом базы данных использовалась программа SQL Server Management Studio фирмы Microsoft. Утилита представляет собой графическую среду, которая позволяет работать с объектами и настройками сервера базы данных.

Функциональность Management Studio может быть расширена специальными дополнениями. Они упрощают процесс создания архитектуры базы данных и написания SQL-скриптов на выборку записей из таблиц. При проектировании базы данных проекта использовались следующие расширения для Management Studio:

1. SQL Prompt, производящий автозаполнение при написании кодов SQL-запросов;
2. SQL Multi Script, позволяющий выполнять один SQL-скрипт на нескольких выбранных базах данных одновременно.

Разработка квалификационной работы проводилась на предприятии, где применяются программы SVN и GIT. Они предназначаются для организации возможности ведения коллективной разработки.

Коллективная разработка создавала дополнительную сложность в проектировании архитектуры источника данных проекта. Она заключалась в том, что у каждого разработчика есть своя база, куда он вносит изменения. После этого программист должен сравнивать изменения, установленные другими разработчиками и разрешать возникающие конфликты. Этот процесс был автоматизирован с применением программного пакета SQL Compare производства компании Redgate Software. Программа позволяет сравнивать структуры нескольких баз данных и формировать искомую схему, в которой учитываются все новые изменения.

Проект подсистемы интеграции обеспечивает обмен данными между «АИС: Объектовый учет» и «Реформа ЖКХ» для нескольких регионов, каждый из которых представлен собственной базой данных. При этом записи таблиц-справочников должны быть идентичными. Для автоматизации проверки эквивалентности этих данных применялся программный продукт SQL Data Compare также производства компании Redgate Software.

Для тестирования процесса взаимодействия с сервисом Реформы использовалась открытая программа soapUI версии 4.5.2. С ее помощью можно отправлять xml-сообщения на указанный адрес сервиса и получать на них ответы.

* 1. Разработка прикладного программного обеспечения

В данном подразделе приводится описание структуры разработанной подсистемы интеграции в целом и ее каждого модуля в отдельности.

* + 1. Структура прикладного программного обеспечения

Подсистема интеграции для федерального портала «Реформа ЖКХ» и «АИС: Объектовый учет» представляет собой приложение со сложной архитектурой. Архитектуру можно подразделить на две смысловые части:

1. подсистема интеграции – это часть проекта, реализованная в виде консольного приложения, обеспечивающая обмен данными с Реформой;
2. панель управления интеграцией – это часть проекта, реализованная в рамках web-приложения «АИС: Объектовый учет». Ее функция заключается в демонстрации хода процесса интеграции данных пользователям управляющих компаний.

На рисунке 5.1 представлена архитектура подсистемы интеграции.



Рисунок 5.1 – Архитектура приложения подсистемы интеграции

Первоначально происходит настройка подсистемы интеграции модулем конфигурации. Он определяет, с какой базой данных и какими дополнительными параметрами должен работать модуль обмена данными.

Затем подсистемы интеграции выполняет ряд следующих действий:

1. формирует запросы на интеграцию данных пользователей управляющих компаний;
2. формирует файловое хранилище документов, которые необходимо загрузить в Реформу.

В процессе интеграции подсистема обращается к базе данных объектового учета при помощи модуля сбора данных. Эти данные формируют объекты указанных классов, которые затем устанавливаются как аргументы API-методов.

Перед отправкой сообщения обрабатываются модулем перехвата, который устанавливает ключ сессии подключения в заголовок каждого xml-документа и пересылает его на сервис федерального портала.

После обработки данных сервис Реформы присылает ответные сообщения, которые анализируются модулем обмена данными. Статус выполнения запроса сохраняется в базу данных объектового учета в таблицу истории его вызова.

На стороне web-приложения реализован модуль панели управления интеграцией. В нем пользователи управляющих компаний имеют возможность наблюдать за ходом процесса интеграции данных их управляющих компаний.

* + 1. Модуль обмена данными

Модуль обмена данными отвечает за пересылку информации о процессах домоуправления организаций на сервис портала «Реформа ЖКХ». Модуль состоит из единственного класса APIProvider, который реализует алгоритм интеграции данных и расширяет возможности стандартных функций API-интерфейса Реформы. Количество значимых строк кода – 589. В таблице 5.1 приводится спецификация класса APIProvider.

Таблица 5.1 – Спецификация класса APIProvider

| Название и тип элемента | Описание |
| --- | --- |
| Глобальные переменные | |
| 1. private DB db | Объект для взаимодействия с данными базы данных как с объектами |
| 2. private DataBinder binder | Объект для обеспечения доступа к методам модуля привязки данных |
| 3. private ApiSoapPortClient client | Объект, при помощи которого осуществляется вызов методов API-интерфейса Реформы |
| 4. private LoginResponse loginResponse | Объект, содержащий информацию о ключе сессии подключения к сервису Реформы |
| 5. private ReloginHelper reloginner | Объект, обеспечивающий доступ к методам модуля бесперебойной передачи данных |
| 6. private bool isSoapFaultMessage | Переменная-флаг, обозначающая статус ответного сообщения |
| 7. private bool isLoginRepeatedly | Переменная-флаг, обозначающая момент выполнения API-метода Login() для получения нового ключа сессии подключения |
| 8. private int? activeQueueId | ID выполняемого интеграционного запроса |
| 9. private string activeMethodName | Наименование выполняемого API-метода |
| 10. private int daysRange = 7 | Переменная устанавливает количество дней, в течение которых от текущей даты модуль формирования запросов идентифицирует изменения пользователей и создает соответствующие интеграционные запросы |
| 11. private AuthHeaderBehavior behavior | Определяет дополнительное «поведение» подключению для создания возможности перехвата сообщений |
| Свойства | |
| 1. public string ActiveMethodName | Свойство доступа к глобальной переменной класса activeMethodName |
| 2. public int? ActiveQueueId | Свойство доступа к глобальной переменной класса activeQueueId |
| Конструкторы | |
| 1. public APIProvider(DB \_db) | \_db – объект для взаимодействия с данными базы данных как с объектами |
| 2. public APIProvider(DB \_db, string \_endPointAddres) | \_db – объект для взаимодействия с данными базы данных как с объектами;  \_endPointAddress – адрес сервиса Реформы |
| Методы | |
| 1. public void CommonInit(DB \_db) | Параметры:\_db – объект для взаимодействия с данными базы.  Назначение:Используется для инициализации общих для всех конструкторов параметров |
| 2. public void HandleFaultReply(Message \_reply) | Параметры: \_reply – ответное xml-сообщение.  Назначение: Callback-метод для обработки приходящего сообщения |
| 3. public void Login() | Назначение: Расширение стандартного API-метода аутентификации на сервисе Реформы |
| 4. public void Logout() | Назначение: Расширение стандартного API-метода закрытия канала обмена данными |
| 5. private void SetRequestForSubmit(IEnumerable<vw\_ReformaActionQueue> \_queueItems) | Параметры: \_queueItems – список запросов организации.  Назначение: Расширение стандартного API-метода подачи заявки на раскрытие данных организацией |
| 6. private void SetNewCompany(IEnumerable<vw\_ReformaActionQueue> \_queueItems) | Параметры: \_queueItems – список запросов организации.  Назначение: Расширение стандартного API-метода создания новой организации |
| 7. private void SetHouseLinkToOrganization(IEnumerable<vw\_ReformaActionQueue> \_queueItems) | Параметры: \_queueItems – список запросов организации.  Назначение: Расширение стандартного API-метода добавления дома в управление организацией |
| 8. private void SetUnlinkFromOrganization(IEnumerable<vw\_ReformaActionQueue> \_queueItems) | Параметры: \_queueItems – список запросов организации.  Назначение: Расширение стандартного API-метода удаления дома из управления организацией |
| 9. public void SetNewHouse(IEnumerable<vw\_ReformaActionQueue> \_queueItems) | Параметры: \_queueItems – список запросов организации.  Назначение: Расширение стандартного API-метода добавления нового дома |
| 10. private void SetHouseProfile(IEnumerable<vw\_ReformaActionQueue> \_queueItems) | Параметры: \_queueItems – список запросов организации.  Назначение: Расширение стандартного API-метода для поставки данных о доме |
| 11. private void SetFileToHouseProfile(IEnumerable<vw\_ReformaActionQueue> \_queueItems) | Параметры: \_queueItems – список запросов организации.  Назначение: Расширение стандартного API-метода для загрузки файла в анкету дома |
| 12. public void SetFileToCompanyProfile(IEnumerable<vw\_ReformaActionQueue> \_queueItems) | Параметры: \_queueItems – список запросов организации.  Назначение: Расширение стандартного API-метода для загрузки файла в анкету организации |
| 13. public void SetFileDeleted(IEnumerable<vw\_ReformaActionQueue> \_queueItems) | Параметры: \_queueItems – список запросов организации.  Назначение: Расширение стандартного API-метода удаления файла |
| 14. private void GetRequestList() | Назначение: Расширение стандартного API-метода для получения и сохранения списка статусов заявок организаций на раскрытие данных |
| 15. private void GetHouseList() | Назначение: Расширение стандартного API-метода для синхронизации объектов, находящихся в управлении организаций |
| 16. private void GetHouseInfo(bool? \_updateAllObjects = false, int? \_orgId = null) | Параметры: \_updateAllObjects – если параметр принимает значение true, тогда осуществляется синхронизация объектов еще не сопоставленных объектов. В противном случае синхронизация происходи для всех объектов.  \_orgId – обозначает ID управляющей организации. При установке данного параметра будет осуществлена синхронизация только тех домов, которые находятся в управлении у данной организации.  Назначение: Расширение стандартного API-метода для синхронизации всех объектов, хранящихся в базах данных обеих ИС |
| 17. private void GetReportingPeriodList() | Назначение: Расширение стандартного API-метода для загрузки/обновления данных справочника об отчетных периодах |
| 18. public void LaunchQueue() | Назначение: Метод обеспечивает процесс обмена данными между системами |

* + 1. Модуль привязки данных

Модуль привязки данных обеспечивает конвертацию SQL-выборки определенной структуры в объект API-класса. Структура модуля состоит из класса DataBinder. Количество значимых строк кода – 52. В таблице 5.2 приводится спецификация класса DataBinder.

Таблица 5.2 – Спецификация класса DataBinder

| Название и тип элемента | Описание |
| --- | --- |
| Глобальные переменные | |
| private char nestedParametersSplitter | Обозначает разделитель вложенных параметров |
| Конструкторы | |
| public DataBinder(char splitter) | Параметры: splitter – символ-разделитель вложенных параметров |
| Методы | |
| public void BindData(ref object \_obj, List<Reforma\_ExtractData\_Result> \_data) | Параметры: \_obj – пустой объект API-класса.  \_data – коллекция данных, полученная в результате выборки данных с помощью вызова хранимой процедуры.  Назначение: Создание объекта API-класса на основе коллекции данных |
| private void SetValue(ref object \_obj, string[] \_path, int \_depth, string \_value, int? \_index, int? \_length) | Параметры: \_obj – объект API-класса.  \_path – наименования уровней вложенности.  \_depth – номер текущего уровня вложенности.  \_value – значение свойства класса.  \_index – индекс элемента коллекции, в который нужно установить значение.  \_length – длина коллекции.  Назначение: Установка значения свойству класса |

* + 1. Модуль сбора данных

Модуль сбора данных производит выборку данных для интеграционного запроса. Работа модуля целиком построена на стороне базы данных, и его структура полностью состоит из иерархически взаимосвязанных хранимых процедур. На рисунке 5.2. представлен механизм очередности вызова процедур в рамках работы модуля.



Рисунок 5.2 – Механизм работы модуля сбора данных

Согласно представленной схеме при сборе данных первоначально осуществляется вызов корневой хранимой процедуры ext.Reforma\_ExtractData. Она определяет структуру возвращаемой выборки данных и создает сложный тип в ORM-системе. Непосредственный сбор информации происходит в остальных хранимых процедурах, выбор которых зависит от наименования API-метода, одного из аргументов корневой процедуры.

Количество значимых строк кода – 1741. В таблице 5.3 представлена спецификация модуля сбора данных.

Таблица 5.3 – Спецификация модуля сбора данных

| Наименование хранимой процедуры | Описание |
| --- | --- |
| ext.Reforma\_ExtractData | Параметры: @objId INT – ID объекта.  @structureId INT – ID структуры объекта.  @orgId INT – ID управляющей организации.  @methodName NVARCHAR(MAX) – наименование API-метода.  Назначение: Корневая хранимая процедура в иерархии выборки данных для дальнейшей поставки в Реформу |
| ext.Reforma\_GetHouseInfo | Параметры: @addressId INT – ID адреса объекта.  Назначение: Обеспечивает получение адресных данных объекта. Возвращаемые данные используются для API-метода GetHouseInfo() |
| ext.Reforma\_SetHouseProfile | Параметры: @objId INT – ID объекта.  @structureId INT – ID структуры объекта.  @orgId INT – ID управляющей организации  @methodName NVARCHAR(MAX) – наименование API-метода.  Назначение: Обеспечивает получение данных из паспорта объекта. Возвращаемые данные используются для API-метода SetHouseProfile() |
| ext.Reforma\_SetNewCompany | Параметры: @orgId INT – ID управляющей организации. @methodName NVARCHAR(MAX) – наименование API-метода.  Назначение: Обеспечивает получение данных для создания новой организации с помощью API-метода SetNewCompany() |
| ext.Reforma\_SetCompanyProfile | Параметры: @orgId INT – ID управляющей организации. @methodName NVARCHAR(MAX) – наименование API-метода.  Назначение: Обеспечивает получение данных для отправки профиля организации с помощью API-метода SetCompanyProfile() |

* + 1. Модуль формирования запросов

Модуль формирования интеграционных запросов пользователей реализован на стороне базы данных. Его структура состоит из трех хранимых процедур: ext.InitQueue, ext.FillActionQueueForOrg, ext.FillActionQueueInGeneral.

Количество значимых строк кода – 213. В таблице 5.4 представлена спецификация модуля формирования запросов.

Таблица 5.4 – Спецификация модуля формирования запросов.

| Наименование хранимой процедуры | Описание |
| --- | --- |
| ext.InitQueue | Параметры: @orgId INT – ID управляющей организации.  Назначение: Формирование группы запросов для первого обмена данными, включая запросы на подачу заявки и регистрацию организации |
| ext.FillActionQueueForOrg | Параметры: @orgId INT – ID адреса объекта.  @startDate DATETIME2 – указывает время, с какого необходимо вести поиск внесенных пользователями изменений.  Назначение: Формирование группы запросов для организации на отправку данных Реформе |
| ext.FillActionQueueInGeneral | Параметры: @startDate DATETIME2 – указывает время, с какого необходимо вести поиск внесенных пользователями изменений  Назначение: Формирование группы общих запросов |

* + 1. Модуль формирования файлового хранилища

Модуль формирования файлового хранилища реализован на стороне базы данных. Его структура состоит из двух хранимых процедур: ext.Reforma\_UpdateFilesRequestsForActionQueue и ext.Reforma\_UpdateFilesStorage.

Количество значимых строк кода – 222. В таблице 5.5 представлена спецификация модуля формирования файлового хранилища.

Таблица 5.5 Спецификация модуля формирования файлового хранилища

|  |  |
| --- | --- |
| Наименование хранимой процедуры | Описание |
| ext.Reforma\_UpdateFilesRequestsForActionQueue | Параметры: @orgId INT – ID организации.  Назначение: Формирование группы интеграционных запросов на сохранение и удаление файлов в базе данных федерального портала |
| ext.Reforma\_UpdateFilesStorage | Параметры: @orgId INT – ID организации.  Назначение: Формирование записей о добавленных организацией файлах в таблице хранилища |

* + 1. Модуль обеспечения бесперебойной передачи данных

Модуль обеспечения бесперебойной передачи данных был разработан с целью продления сессии подключения. Согласно регламенту взаимодействия с сервисом Реформы время пересылки данных в рамках одной сессии ограничено до 10 минут. Модуль производит переподключение к сервису каждые 8 минут для устранения возникновения исключительных ситуаций.

Структура модуля состоит из класса ReloginHelper. Количество значимых строк кода модуля – 8. В таблице 5.6 приведена спецификация класса ReloginHelper.

Таблица 5.6 – Спецификация класса ReloginHelper

| Название и тип элемента | Описание |
| --- | --- |
| Глобальные переменные | |
| private APIProvider provider | Объект класса модуля обеспечения обмена данными (APIProvider) |
| private DateTime sessionStart | Время начала поставки данных |
| Свойства | |
| public bool IsSessionExpired | Свойство, определяющее закончилось ли время подключения |
| Конструкторы | |
| public ReloginHelper(APIProvider provider) | Параметры: provider – объект класса модуля обеспечения обмена данными (APIProvider) |
| Методы | |
| public void StartTimer() | Назначение: Производит запуск таймера, отмеряющего время подключения |
| public void Relogin() | Назначение: Производит переподключение к сервису Реформу с помощью API-методов аутентификации |

* + 1. Вспомогательный модуль

Вспомогательный модуль реализован в виде статического класса Servant. Класс содержит набор методов для обеспечения поддержки интеграционного процесса.

Количество значимых строк кода – 92. В таблице 5.7 представлена спецификация класса Servant.

Таблица 5.7 – Спецификация класса Servant

| Название и тип элемента | Описание |
| --- | --- |
| Методы | |
| public static int GetRequiredObjectId(DB \_db, HouseData \_address) | Параметры: \_db – объект для взаимодействия с данными базы данных как с объектами.  \_address – API-объект, обозначающий адрес дома.  Назначение: Получение ID объекта в базе данных ОУ по адресу из Реформы |
| public static int[] GetRequiredObjectTypes(DB \_db) | Параметры: \_db – объект для взаимодействия с данными базы данных как с объектами.  Назначение: Получение идентификаторов типов объектов |
| public static IEnumerable<vw\_ReformaActionQueue> FilterQueueActionsByMethodName(DB \_db, IEnumerable<vw\_ReformaActionQueue> \_actions, string \_methodName) | Параметры: \_db – объект для взаимодействия с данными базы данных как с объектами.  \_actions – коллекция запросов.  \_methodName – наименование API-метода.  Назначение: Получение информации об определенных запросах из общей коллекции |
| public static FileObject EncodeFileContent(DB \_db, vw\_ReformaFilesStorage \_doc) | Параметры: \_db – объект для взаимодействия с данными базы данных как с объектами.  \_doc – коллекция запросов.  Назначение: Создание объекта для интеграции данных о файле |
| public static bool IsAllActionsSucceed(IEnumerable<vw\_ReformaActionQueue> \_actions, string \_methodName) | Параметры: \_actions – коллекция запросов.  \_methodName – наименование API-метода.  Назначение: Проверяет выполнены ли успешно все методы определенного типа |
| public static bool IsBidApproved(DB \_db, int \_orgId) | Параметры: \_db – объект для взаимодействия с данными базы данных как с объектами.  \_orgId – ID организации.  Назначение: Проверяет подтверждена ли заявка на раскрытие данных организацией |
| public static IEnumerable<vw\_cmn\_Organization> GetSynchronizedOrgs(DB \_db) | Параметры: \_db – объект для взаимодействия с данными базы данных как с объектами.  Назначение: Получение списка организаций, осуществляющих интеграционный обмен данными |
| public static void SaveLog(DB \_db, string \_soapReply, string \_soapRequest, int \_activeActionId, bool \_isError, int? \_code = null, string \_message = "") | Параметры: \_db – объект для взаимодействия с данными базы данных как с объектами.  \_soapReply – xml-сообщение ответа от Реформы.  \_soapRequest – xml-сообщение запроса.  \_activeActionId – ID выполненного запроса.  \_isError – обозначает статус выполнения запроса.  \_code – код сообщения.  \_message – текст ответного сообщения.  Назначение: Сохранение истории выполнения запроса |

* + 1. Модуль перехвата сообщений

Модуль перехвата сообщений был разработан с целью установки ключа сессии подключения в заголовки отправляемых сообщений и для их сохранения.

Структура модуля состоит из класса AuthHeaderBehavior, реализующего интерфейсы IEndpointBehavior и IClientMessageInspector. Количество значимых строк кода составляет 22. В таблице 5.8 представлена спецификация класса AuthHeaderBehavior.

Таблица 5.8 – Спецификация класса AuthHeaderBehavior

| Название и тип элемента | Описание |
| --- | --- |
| Свойства | |
| public string SoapReply | Хранит ответное сообщение от Реформы |
| public string SoapRequest | Хранит xml-сообщение запроса |
| public TokenProvider TokenProvider | Объект класса используется для хранения ключа сессии |
| public APIProvider ApiProvider | Объект класса модуля обеспечения обмена данными (APIProvider) |
| Конструкторы | |
| public AuthHeaderBehavior() | Инициализирует экземпляр класса |
| public void AfterReceiveReply(ref System.ServiceModel.Channels.Message reply, object correlationState) | Параметры: reply – ответное xml-сообщение, приходящие от сервиса Реформы.  correlationState – данные состояния корреляции.  Назначение: Метод срабатывает в момент прихода ответного xml-сообщения и вызывает callback-функцию на стороне модуля обеспечения обмена данными для сохранения истории запроса |
| public object BeforeSendRequest(ref System.ServiceModel.Channels.Message request, System.ServiceModel.IClientChannel channel) | Параметры: request – ответное xml-сообщение, приходящие от сервиса Реформы.  channel – канал объекта клиента WCF.  Назначение: Метод срабатывает в момент отправки xml-сообщения и устанавливает ключ сессии подключения в заголовок сообщения |

* + 1. Модуль конфигурации

Назначение модуля конфигурации заключается в настройке подсистемы интеграции для работы с несколькими базами данных. Структура модуля состоит из классов CommonConfiguration и RegionConfigurationHandler. Статический класс CommonConfiguration описывает основные настройки конфигурации. Класс RegionConfigurationHandler обеспечивает доступ к настройкам, хранящимся в xml-файле конфигурации. Количество значимых строк кода модуля – 44.

В таблицах 5.9 и 5.10 представлены спецификации классов CommonConfiguration и RegionConfigurationHandler соответственно.

Таблица 5.9 – Спецификация класса CommonConfiguration

| Название и тип элемента | Описание |
| --- | --- |
| Свойства | |
| public static string regionName | Название региона |
| public static string connectionString | Строка подключения к базе данных |
| public static string people\_site\_root\_url | URL-адрес папки с файлами на сервере |
| public static string people\_site\_path | Путь до папки с файлами на сервере |
| Методы | |
| public static void DropSettings() | Назначение: Обеспечивает сброс настроек конфигурации |

Таблица 5.10 – Спецификация класса RegionConfigurationHandler

| Название и тип элемента | Описание |
| --- | --- |
| Глобальные переменные | |
| private string configFilePath | Обозначает имя xml-файла конфигурации. Значение по умолчанию: @"\RegionsConfig.xml" |
| private List<string> handledRegions | Список наименований регионов, с которыми подсистема интеграции будет работать |
| Конструкторы | |
| public RegionsConfigurationHandler() | Инициализирует экземпляр класса |
| Методы | |
| public bool SetConfiguration() | Назначение: Загрузка настроек конфигурации из файла и их установка |

* + 1. Модуль панели управления интеграцией

Для предоставления возможности пользователям управляющих компаний наблюдать процесс интеграции данных, была разработана специальная панель управления. Она поддерживает следующую функциональность:

1. подача заявки организацией на раскрытие данных;
2. отображение списка запросов и истории их выполнения;
3. отображение статистики интеграции.

Панель интеграции реализована как часть web-приложения. Количество значимых строк контроллера составляет 342. В таблице 5.11 представлена спецификация контроллера ReformIntegrationController.

Таблица 5.11 – Спецификация контроллера ReformIntegrationController.

| Название и тип элемента | Описание |
| --- | --- |
| Действия (Actions) | |
| RegisterOrganization(int orgId) | Параметры: orgID – ID управляющей организации.  Назначение: Метод срабатывает, когда пользователь УК подает заявку на раскрытие данных, и осуществляет вызов хранимой процедуры ext.InitQueue |
| public ActionResult Index() | Назначение: Отображение главной страницы панели интеграции |
| public ActionResult List() | Назначение: Отображение списка интеграционных запросов |
| public ActionResult HistoryList(int id) | Параметры: id – ID запроса.  Назначение: Отображение истории выполнения запроса |
| public ActionResult Stats() | Назначение: Отображение статистики интеграционного процесса УК |

* 1. Особенности эксплуатации и сопровождения системы

В процессе эксплуатации сборка программы работает на компьютере-сервере с установленной операционной системой Windows Server 2008 R2 Standard. Вызов приложения на исполнение должно происходить автоматически, для этого настройка запуска программы осуществляется в меню диспетчера сервера, представленного на рисунке 5.3.



Рисунок 5.3 – Диспетчер сервера

Чтобы не влиять на скорость работы web-приложения «АИС: Объектовый учет», подсистема интеграции настроена на выполнение процесса обмена данными в ночное время, когда с системой объектового учета взаимодействует минимальное число пользователей.

Информационные системы для сферы ЖКХ быстро меняются под влиянием поступающих требований от заказчиков или правительства. Подсистема интеграции устроена таким образом, чтобы при переходе к новой версии максимально минимизировать затраты времени. Выпуск новой версии программы для обмена данными возможен при возникновении следующих ситуаций:

1. Реформа изменила API-интерфейс. В этом случае доработка подсистемы интеграции сводится к адаптации модулей сбора данных и обмена данными для обеспечения поддержки новых возможностей API-интерфейса;
2. Реформа добавила новые параметры интеграции. В этом случае доработка подсистемы интеграции сводится к адаптации модуля сбора данных и определению новых параметров в таблице-справочнике.
   1. Интерфейс пользователя с системой

В данном подразделе рассматриваются возможности пользовательского интерфейса панели управления интеграцией, процесс установки для использования пользовательской части приложения интеграции и обработка возникающих в процессе работы web-приложения исключительных ситуаций.

* + 1. Руководство пользователя

Интерфейс панели управления интеграцией состоит из четырех html-страниц:

1. главная страница;
2. страница со списком запросов организации;
3. страница с историей вызова конкретного запроса;
4. страница отображения статистики процесса интеграции.

Чтобы попасть на главную страницу пользователю управляющей компании необходимо авторизоваться в системе и зайти в личный кабинет. На рисунке 5.4 отображено положение меню доступа на главную страницу панели управления интеграцией.



Рисунок 5.4 – Положение меню доступа на главную страницу интеграции

После перехода по ссылке пользователь попадает на главную страницу панели управления, представленную на рисунке 5.5.



Рисунок 5.5 – Главная страница панели управления интеграцией

На главной странице пользователь УК может прочитать правила интеграции данных и подать заявку на раскрытие данных своей организации.

После нажатия на кнопку о подаче заявки пользователь увидит диалоговое окно, представленное на рисунке 5.6.



Рисунок 5.6 – Диалоговое окно подачи заявки на раскрытие данных

Диалоговое окно создано для подтверждения процесса подачи пользователем заявки, после которого статус главной страницы панели управления изменится, как указано на рисунке 5.7.



Рисунок 5.7 – Изменение статуса заявки после ее подачи

После рассмотрения и подтверждения заявки администратором портала, интерфейс главной страницы будет выглядеть, как представлено на рисунке 5.8.



Рисунок 5.8 – Интерфейс панели управления после подтверждения заявки

В случае отклонения заявки на раскрытие данных, статус на главной странице также оповестит об этом пользователя. Вариант построения интерфейса в этом случае представлен на рисунке 5.9.



Рисунок 5.9 – Статус отклонения заявки

Подменю «Управление интеграцией» позволяет пользователю УК ознакомиться со списком интеграционных запросов, с помощью которых осуществляется раскрытие данных организации. Страница списка интеграционных запросов показана на рисунке 5.10.



Рисунок 5.10 – Страница для отображения списка интеграционных запросов

Для удобства просмотра списка на странице есть фильтр, позволяющий отображать только определенные виды запросов. Для смены отображения групп и количества запросов пользователь может использовать встроенную пагинацию.

Для просмотра страницы истории вызова определенного запроса пользователь должен перейти по ссылке, которая представляет собой иконку «блокнота» и находится слева от названия интеграционного запроса в списке.

На рисунке 5.11 отображена страница истории вызова запроса.



Рисунок 5.11 – Страница отображения истории вызова запроса

История вызова содержит записи о датах выполнения запроса и статусах их выполнения, а также информации об ошибках или предупреждениях, возникших при пересылке данных.

Страница статистики интеграции позволяет наглядно демонстрировать пользователям УК процесс интеграции их данных. На рисунке 5.12 отображена страница статистики интеграции.



Рисунок 5.12 – Страница отображения статистики интеграции

Всего на странице представлено несколько графиков, каждый из которых отражает определенную характеристику процесса интеграции. График синхронизации объектов показывает соотношение сопоставленных объектов для «АИС: Объектовый учет» и «Реформа ЖКХ», находящихся в управлении организации. На рисунке 5.13 отображен рассматриваемый график.



Рисунок 5.13 – График синхронизации объектов

Следующий за ним график отражает активность организации за последние семь дней. Он показывает количество выполненных успешно и невыполненных интеграционных запросов по дням. На рисунке 5.14 отображен рассматриваемый график.



Рисунок 5.14 – График активности управляющей организации

График, представленный на рисунке 5.15, показывает общую статистику количества отправленных запросов по дням для всех организаций. Временной интервал для отображения данных – последние семь дней.



Рисунок 5.15 – Статистика интеграции для организаций.

* + 1. Инсталляция и настройка

Для использования подсистемы интеграции пользователям управляющих компаний необходимо зарегистрировать организацию в системе «АИС: Объектовый учет». Чтобы начать процесс раскрытия данных на сайте портала «Реформа ЖКХ», необходимо подать заявку и дождаться ее подтверждения. Никаких других дополнительных действий от пользователя не требуется, так как процесс интеграции данных полностью автоматизирован.

Взаимодействие с «АИС: Объектовый учет» возможно при наличии интернет соединения и любого установленного браузера, включая Internet Explorer 9 версии и выше.

* + 1. Исключительные ситуации и их обработка

Модуль пользовательского интерфейса разработан с учетом обработки всех возможных исключительных ситуаций.

Другая часть проекта, которая обеспечивает обмен данными между ИС, также обрабатывает все возможные исключительные ситуации, отображая текст возникающих ошибок в консоль.

Заключение

# Список литературы

1. Гамма, Э. Приемы объектно-ориентированного программирования. Паттерны проектирования / Э. Гамма, Р. Хелм, Р. Джонсон, Дж. Влиссидес. – СПб. : Питер, 2001. – 344 с.
2. Гончар, С.Т. Безопасность и экологичность объекта проектирования: учебное пособие по дипломному проектированию / С.Т. Гончар. – 2-е изд. – Ульяновск : УлГТУ, 2009. – 165 с.
3. ГОСТ 19.701-90. Единая система конструкторской документации. Схемы алгоритмов, программ, данных и систем. Условные обозначения и правила выполнения. – М. : Стандартинформ, 2010. – 8 с.
4. ГОСТ 2.105-95. Единая система конструкторской документации. Общие требования к текстовым документам. – М. : Стандартинформ, 1996. – 9 с.
5. Постановление правительства Российской федерации № 731 от 23 сентября 2010 г. «Об утверждении стандарта раскрытия информации организациями, осуществляющими деятельность в сфере управления многоквартирными домами» [Электронный ресурс] / КонсультантПлюс. –

[Б. м. : б. и.], 2010. –

Режим доступа: <http://www.consultant.ru/document/cons_doc_LAW_169248/> (дата обращения: 12.05.2015).

1. Родионов В.В. Дипломное проектирование: учебно –методическое пособие для студентов специальности 23020165 «Информационные системы и технологии» / В.В. Родионов. – Ульяновск : УлГТУ, 2008. – 98 с.
2. Троелсен, Э. Язык программирования C# 2010 и платформа .NET 4 / Э. Троелсен. – 5-е изд. - М. : Вильямс, 2010. – 1392 с.
3. Фленов, М. Е. Библия C# / М.Е.Фленов. – 2-е изд. – СПб. : БХВ-Петербург, 2011. – 560 с.
4. Cheng, S. Microsoft Windows Communication Foundation 4.0 Cookbook for developing SOA Applications / S.Cheng. – Birmingham : Packt Publishing, 2010. – 316 pp.
5. Lowy, J. Programming WCF Services / J. Lowy. – NY : O’Reilly – 2010. – 910 pp.
6. Mukherjee, S. Thinking in LINQ / S. Mukherjee. – NY : Appress, 2015. – 259 pp.
7. Palermo, J. ASP.NET MVC 4 in Action / J.Palermo, J. Bogard, E. Hexter, M. Hinze, J. Skinner. – Shelter Island, NY : Manning, 2012. – 440 pp.
8. Skeet, J. C# in depth. Third edition / J.Skeet. – Shelter Island, NY : Manning, 2014. – 614 pp.

# Приложение А

(обязательное)

Текст программы

**Модуль обмена данными**

using System;

using System.Linq;