МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ

РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение

высшего образования

«УЛЬЯНОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информационных систем и технологий

Кафедра Информационные системы

Отчет по практике защищен с оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(оценка прописью)

Руководитель практики от университета \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(должность)  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись) (ФИО)

**ОТЧЕТ по** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ практике

(вид практики)

Студента \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(ФИО)

Направление (специальность, профиль) подготовки \_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Группа \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Место прохождения практики \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(наименование профильной организации, подразделение университета)

Ульяновск

20\_\_\_ г.

**Содержание**

[Введение 3](#_Toc30838510)

[1. Постановка задачи 5](#_Toc30838511)

[2. Системная модель объекта исследования. 6](#_Toc30838512)

[2.1. Концептуальная архитектура системы 6](#_Toc30838513)

[2.2. Теоретико-множественное описание 6](#_Toc30838514)

[2.3. Проектирование и описание системы хранения данных и систем знаний. 7](#_Toc30838515)

[3. Проектирование системы 9](#_Toc30838516)

[3.1. Техническое задание 9](#_Toc30838517)

[1. Общие сведения 9](#_Toc30838518)

[2. Назначение и цели создания системы: 9](#_Toc30838519)

[3 Характеристика объекта автоматизации 10](#_Toc30838520)

[4. Требования к системе 10](#_Toc30838521)

[4.3. Требования к видам обеспечения: 12](#_Toc30838522)

[5. Состав и содержание работ по созданию системы. 12](#_Toc30838523)

[6 Порядок контроля и приемки системы 13](#_Toc30838524)

[7. Требования к составу и содержанию работ по подготовке объекта автоматизации к вводу системы в действие 13](#_Toc30838525)

[3.2. Диаграмма вариантов использования 13](#_Toc30838526)

[3.3. Диаграмма классов 14](#_Toc30838527)

[3.4. Диаграмма состояний 15](#_Toc30838528)

[3.5. Диаграмма последовательности 16](#_Toc30838529)

[4. Прототип ПО 18](#_Toc30838530)

[4.1. Описание используемых компонент системы 18](#_Toc30838531)

[4.2. Реализация ПО 21](#_Toc30838532)

[Заключение 23](#_Toc30838533)

[Библиографический список 24](#_Toc30838534)

# Введение

В настоящее время предметом исследований большинства современных разработчиков информационных систем являются методы и подходы, чтобы построить не просто распределенные информационные системы, но и интеллектуальные, предусматривающие возможность их адаптации к конкретной предметной области. Подобная потребность появилась при рассмотрении распределенной системы отслеживания задач по технике Помодоро, чтобы исключить необходимость в роли человека, распределяющего задачи, и автоматизировать этот процесс оптимально.

Существует множество инструментов для исследований и разработки таких систем, анализа больших массивов данных с учетом специфики предметной области. Одной из наиболее универсальных методик представления экспертных знаний, в данном случае знаний при распределении заданий по исполнителям, полностью описывающие семантически, является онтологический подход.

Актуальность применения онтологий для реализации менеджмента описана в статье «Онтология для управления проектами: обследование» [[14](#_fbqlbsjopug9)], где представлен систематический обзор литературы, посвященный онтологиям управления проектами. Автор выявил отсутствие стандартизации в терминологии и концепциях, отсутствие систематического моделирования областей и использования онтологий, главным образом, в прототипных системах онтологий, которые затрагивают довольно ограниченные аспекты процессов управления.

Цель работы: разработать и исследовать систему распределения заданий по исполнителям с применением онтологии.

Задачи исследовательской работы:

* изучить существующие решения распределения в тайм-менеджменте;
* рассмотреть применение онтологии в управлении задачами и проектами;
* разработать собственную реализацию алгоритма распределения;
* оценить эффективность и применимость разработанного алгоритма.

Объектом исследования является открытая база данных отслеживания ошибок (баг-трекера) Redmine.

Предметом исследования являются инструменты онтологии прикладной области и методы классификации.

План работы:

* Провести анализ прикладной области IT-компании, описать обзор применения онтологии в бизнес-процессах;
* Изучить методы и способы распределения задач по исполнителям, выявить особенности;
* Спроектировать автоматизированную систему (Создать диаграммы IDEF0, диаграммы UML, Техническое задание);
* Провести предобработку данных, представления данных в статическом и динамическом виде;
* Построить онтологию ПрО задач в компании;
* Разработать интерфейс оптимального распределения задач по исполнителям;
* Разработать и протестировать программную систему распределения задач;
* Создать документацию ПО (инструкция пользователя).

# Постановка задачи

Для постановки задачи стоит определиться с основными объектами создаваемой системы и их признаками.

* внутренние задачи IT-компании
  + область применения (домен),
  + длительность исполнения (время),
  + оценка сложности задания
* сотрудники компании
  + области специализации (домен),
  + время занятости,
  + уровень квалификации.

Необходимо оптимально сопоставить признаки задач соответственно признакам сотрудников с применением онтологии.

Данная система предназначается для предприятий IT области с большим объемом проектов. Она является частью системы управления задачами, ведет учет и распределение заданий на основе ограничений онтологии ПрО между пользователями, объединенными в группу.

Данные для онтологии используются из Административной БД сайта Redmine. БД сайта содержит задания по ошибкам (багам) во время разработки их веб-проекта.

# Системная модель объекта исследования.

## Концептуальная архитектура системы

Архитектура системы как модуль поведения описывается в нотации IDEF0. Такое функциональное описание бизнес-процессов представлено на рис.2.1.

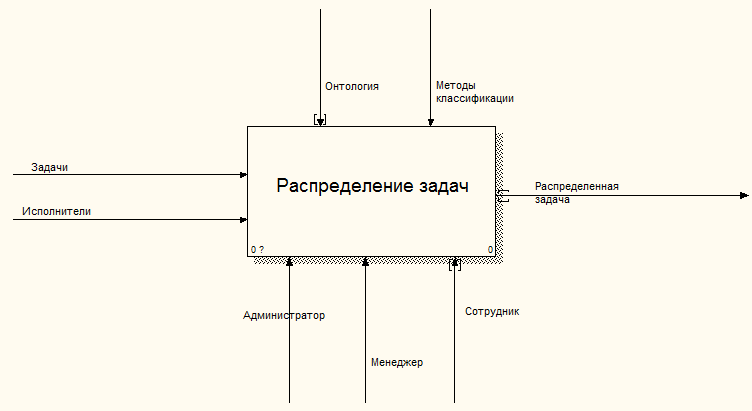


Рис.2.1 Диаграмма архитектуры системы

На вход получаем данные задачи и исполнителей. Они преобразуются с управляющими элементами: инструменты. Архитектура “черный ящик” снизу управляется пользователями. Подробнее рассмотрено в следующем этапе.

## Теоретико-множественное описание

Для задачи распределения будут использоваться инструменты онтологии.

Задачи будут иметь структуру:

Z = <D, T, P> при Z = {z1,... ,zn}, n–количество заданий

где

D – множество доменов, к которым относится задание (продукты компании, компоненты);

Т = {T0, Tend} – множество длительности исполнения задания, где

T0 – множество дат при постановке задания на исполнение,

Tend – множество дат после выполнения задания;

Р – множество оценок сложности задания.

Исполнители (assignee) будут иметь следующую структуру:

A = <D, T, P> при A = {a1,... ,am}, m–количество исполнителей,

где

D – множество области специализации (продукты компании, компоненты, аналогично);

Т = {T0, Tend} – множество длительности занятости исполнителя, где

T0 – множество дат начала занятости,

Tend – множество дат после выполнения задания;

Р – множество уровня квалификации.

## Проектирование и описание системы хранения данных и систем знаний.

Источники с исходными данными будут, как формироваться вручную, так и извлекаться в виде csv, xml формата представления из открытой базы данных сайта Redmine (рис.4.1).

Хранилище знаний – онтология в формате OWL. Онтология будет построена вручную с учетом малого опыта в использовании редакторов онтологий автором проекта.

В дальнейшем предполагается использовать другие способы ее построения:

* автоматизированное, где синтаксический анализ разбивается на несколько шагов:

1. выделение лексем;
2. морфологический анализ;
3. выделение простых вспомогательных структур;
4. выделение структур согласования, управления и объектов;
5. выделение набора жестких структур, описывающих отношения между понятиями.

* автоматическое аннотирование с JAVA.

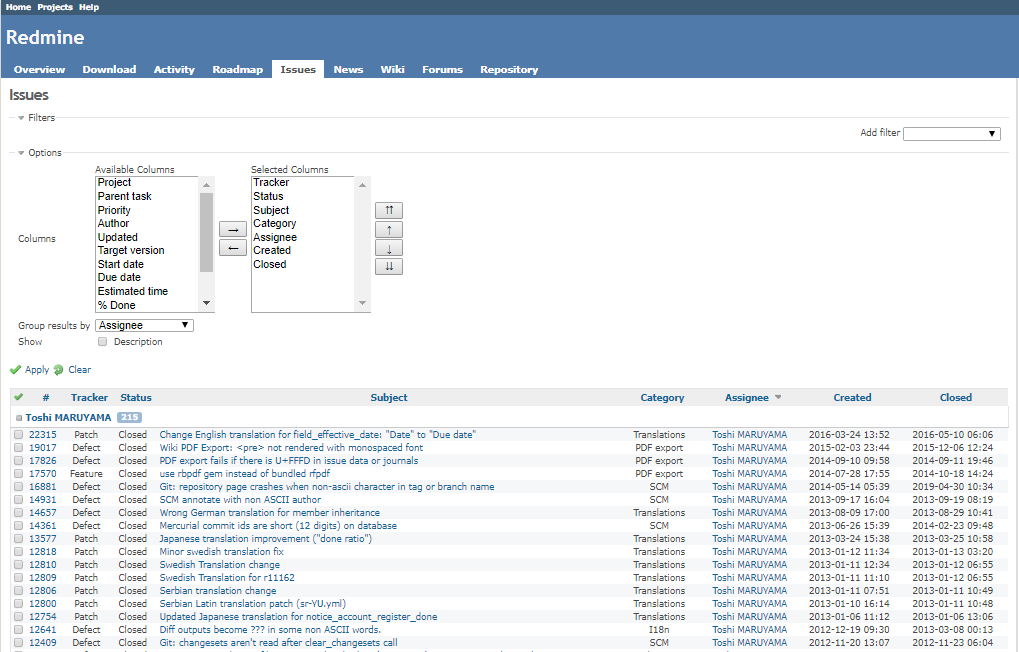


Рис.4.1. Страница отчетов сайта Redmine

Создание онтологий задач IT-специалистов вручную предполагает создание классов и добавлении ограничений.

# Проектирование системы

## Техническое задание

### 1. Общие сведения

1.1. Полное наименование системы: Автоматизированная система распределений заданий по исполнителям на основе онтологии.

Условное обозначение системы: АИС «TasktrackerOnto».

1.2. Договор не заключался

1.3. Наименование предприятий разработчика: Иванова Н.П.

Наименование заказчика системы, их реквизиты: отсутствуют.

1.4. Перечень документов, на основании которых создается ИС: Техническое задание, задание к курсовому проекту.

1.5. Плановые сроки начала работ – начало семестра, и окончания работ – конец семестра.

1.6. Сведений об источниках и порядке финансирования работ нет, так как работы не финансируются.

1.7. Порядок оформления и предъявления заказчику результатов работ по созданию системы, её частей и отдельных средств не предъявляется.

### 2. Назначение и цели создания системы:

2.1. Разрабатываемая система предназначена для автоматизации работы менеджера при распределении заданий исполнителям с применением онтологии.

2.2. Цель создания системы: уменьшить трудоемкость системы управления задачами (распределить, выполнить) для сотрудников IT-компании.

### 3 Характеристика объекта автоматизации

3.1 Краткие сведения об объекте автоматизации

Данная система может использоваться для предприятий малого бизнеса в IT области. Она является частью системы управления задачами, ведет учет и распределение заданий на основе ограничений онтологии ПрО между пользователями, объединенными в группу.

Данные для онтологии используются из Административной БД сайта Redmine. БД сайта содержит задания по ошибкам (багам) во время разработки их веб-проекта.

3.2 Сведения об условиях эксплуатации и характеристиках окружающей среды:

– бесперебойная подача питания;

– защита от несанкционированного воздействия;

– доступ к сети Интернет (желательно).

Требования к процессам взаимодействия с ИС делается в соответствии со стандартами и законами.

### 4. Требования к системе

4.1. Требования к системе в целом

Требования к структуре и функционированию полноценной системы:

* подсистема распределения задач (распределить задачу по исполнителям, проверить ограничения по онтологии)
* подсистема выполнения задачи (отображение списка задач, выбор задачи из соответствующего списка, уведомление об исполнении задачи);
* подсистема хранилища (данные о пользователях и задачах и их взаимосвязи в виде свойств).

Описанные выше последние два пункта будут реализованы в данном проекте.

Требования к персоналу: владеть навыками пользования ПК и браузера на базовом уровне.

Требования к надёжности:

* обеспечить контроль и обработку ошибок пользователя;
* обеспечить возможность восстановления данных с внешнего накопителя после восстановления активного накопителя;
* БД системы необходимо резервировать минимум раз в месяц.

Требования к эргономике и технической эстетике:

* интерфейс системы должен быть удобным и простым;
* пакет программного обеспечения включает все необходимые библиотеки.

4.2. Требования к функциям:

* Учёт существующих задач: Менеджер может изменять список текущих задач организации (добавление, удаление). Необходимо обеспечить корректный отклик на соответствующий запрос.

Входные данные: список задач.

В результате на странице должны отображаться задачи с отметкой статуса исполнения, который определяется пользователем.

Действующие средства сбора, передачи и обработки информации: информация о текущих задачах заносится администратором в БД и передаётся исполнителям.

* Распределения актуальных задач среди пользователей: Формирование индивидов для объектов «задача», «организация», «сотрудник» и формирование свойств типов данных и объектных свойств для распределения актуальных задач.

Входные данные: класс задач и класс исполнителей.

В результате ограничений онтологии задач выявляют актуальные задачи, задачи со статусом исполнения.

Действующие средства сбора, передачи и обработки информации: информация о текущих задачах заносится администратором в БД и передаётся исполнителям.

### 4.3. Требования к видам обеспечения:

* Требования к информационному обеспечению: данные хранить в онтологии ПрО, реализовать многопользовательский режим доступа к данным.
* Требования к алгоритмическому обеспечению: разработать алгоритмы для суперпользователя, для пользователя с низкими правами доступа работа не меняется.

Требования к программному обеспечению: поддержка ПО в любой операционной системе; возможность визуализации результатов в виде таблиц.

* Требования к лингвистическому обеспечению: язык интерфейса – русский, язык программирования – python.
* Требования к техническому обеспечению: Процессор – Intel(R) Core 1.5ГГц и выше, Объём оперативной памяти – 1 Гб и выше,   
  Объём HDD – 20 Гб и выше.

### 5. Состав и содержание работ по созданию системы.

5.1. Перечень стадий и этапов работ, сроки исполнения ведутся соответственно ГОСТ 34.201.

5.2. Вид и порядок проведения экспертизы технической документации выявить в ходе лабораторных работ по подготовке курсового проекта.

5.3 Программа обеспечения надежности предусматривает на стадиях разработки, изготовления и эксплуатации контроль частого резервного копирования проекта.

### 6 Порядок контроля и приемки системы

Не предъявляется.

### 7. Требования к составу и содержанию работ по подготовке объекта автоматизации к вводу системы в действие

7.1 Преобразование входной информации к машиночитаемому виду путем ввода данных с клавиатуры.

7.2 Сроки и порядок комплектования и обучения персонала в течение недели до опытной эксплуатации. За это время необходимо изучить руководство пользователя и практически освоить работу Системы.

## Диаграмма вариантов использования

Диаграмма вариантов использования (рис.3.1) описывает то, что система будет делать в процессе ее функционирования. Она отображает работу следующих актёров[[1]](#footnote-1):

* сотрудник, который обладает прецедентами подсистемы выполнения задачи, которая в свою очередь связывается с подсистемой отсчета времени;
* менеджер, который в подсистеме статистики обладает прецедентом, имеющий связь расширения (extend) из подсистемы выполнения задач, чтобы иметь данные статуса задач. Также он имеет прецеденты подсистемы управления задачами, чтобы непосредственно эти задачи были созданы.

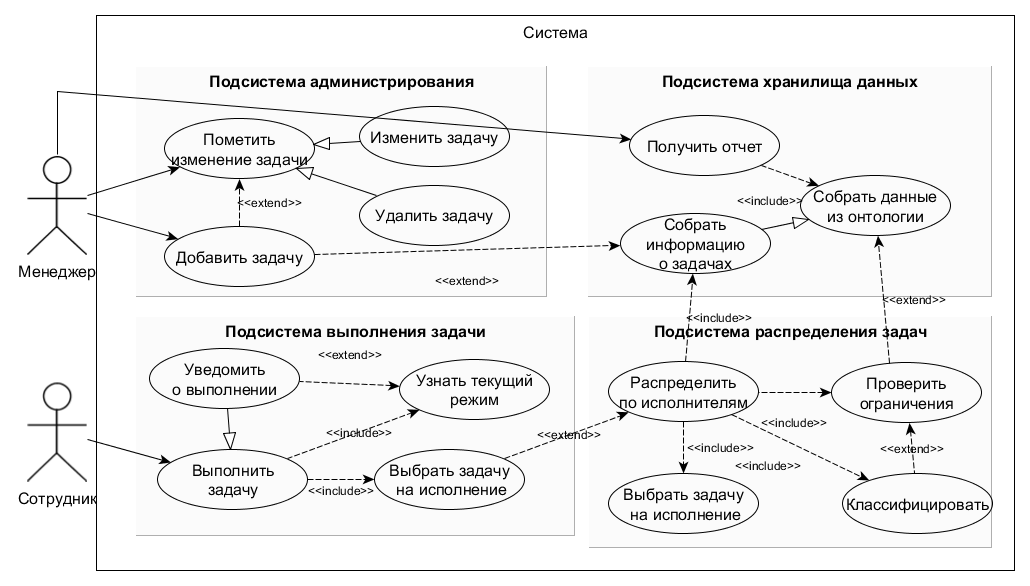


Рис. 3.1 UML-диаграмма прецедентов системы

## Диаграмма классов

Диаграмма классов детально описывает структуру модели системы с использованием терминологии классов объектно-ориентированного программирования. В данном случае в диаграмме (рис.3.2) указано отношение композиции для пользователей (Users) для сотрудников (Workers) и руководителей (Organisators). Это отношение служит для выделения специальной формы отношения "часть-целое", при которой составляющие части в некотором смысле находятся внутри целого.

Пользователь должен быть авторизован, чтобы иметь доступ к классам задач (Tasks) и отделы (Departments), для этого используется отношение зависимости с проверкой разрешения (CheckPerms) к этим классам. Такие отношения чаще всего использованы в данной диаграмме.

Отношение обобщения описывает иерархическое строение классов и наследование их свойств и поведения. Так, класс выполнения задач (Doing\_Task) является потомком класса задач (Task), который наследует его свойства, которые являются открытыми, изображенные знаком «+».

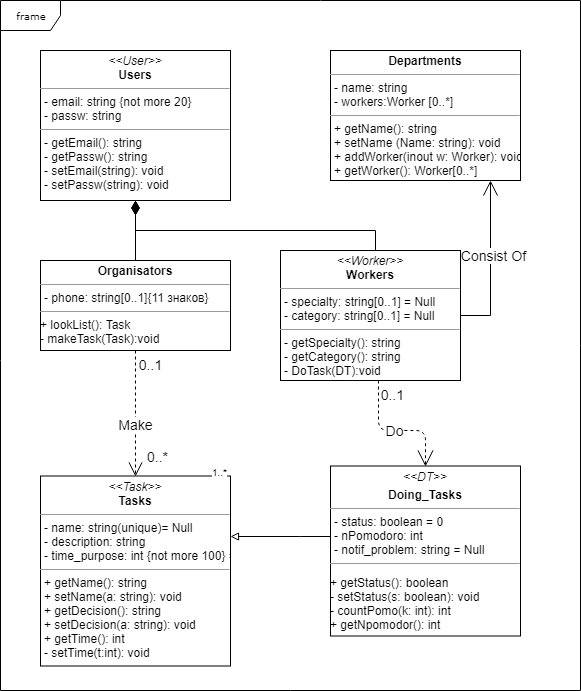


Рис.3.2 Диаграмма классов UML

## Диаграмма состояний

Диаграмма состояний представляет собой состояние, соединенные переходами, характеризует жизненный цикл, который начинается при запуске. Диаграмма (рис.3.3) описывает работу распределения задачи с учетом создаваемой системы. Указаны роли «Менеджер» и «Сотрудник». «Система» выполняет большую часть задач, а значит, автоматизирует работу обеих сторон.

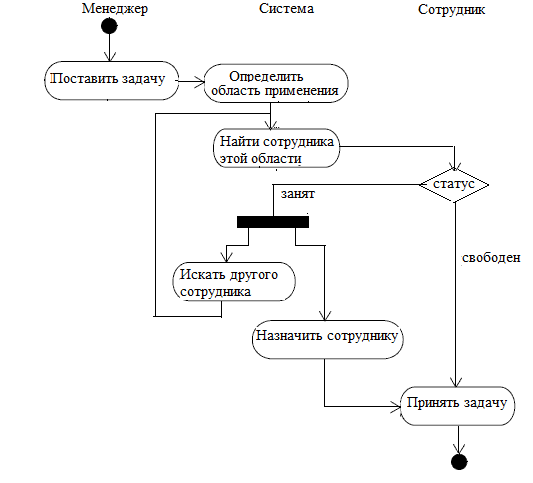


Рис.3.3 Диаграмма состояний

## Диаграмма последовательности

Диаграмма последовательности (рис.3.4) отражает временные особенности передачи сообщений между частями программы, а также показывает время жизни классов, которые представлены в диаграмме классов (рис.3.2).

В данной диаграмме частями программы являются основные модули программной разработки: главный модуль, объединяющий в себе работу модуля связи с онтологией и модуля классификации, оптимизирующего распределение. В рамках данного курсового проекта эта часть не будет реализована.

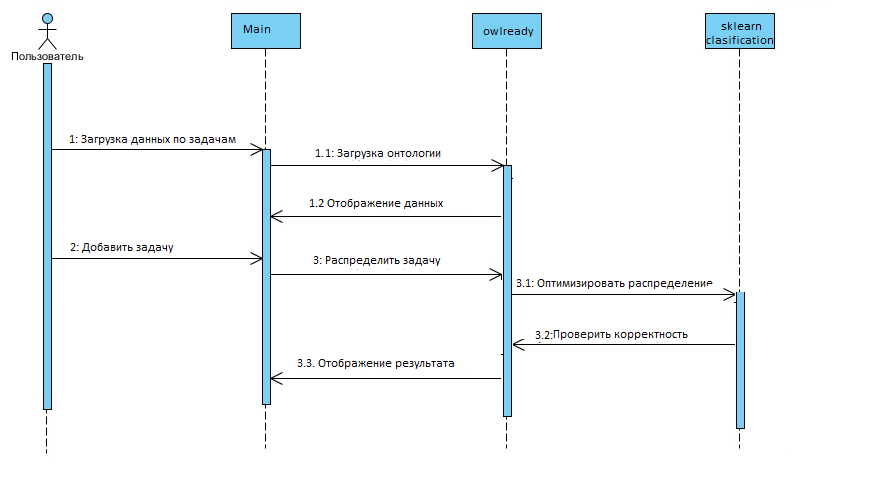


Рис.3.4 Диаграмма последовательности

# Прототип ПО

На вход будет поступать новая задача и информация о ней соответственно кортежу, описанному в следующем пункте. В результате работы онтологии будет выбран исполнитель.

Реализация в виде desktop-приложения, иначе называемых «оконными» приложениями. В них используется графический интерфейс и ввод информации осуществляется при помощи клавиатуры и мыши. Данные приложения позволяют создавать практически любые по сложности решения.

Онтология в формате OWL представлена в следующем пункте.

Поддержка любой ОС реализуется с помощью конвертации формата Python в любой исполняемый файл. Для этого используется модуль используемого в работе языка программирования.

## Описание используемых компонент системы

редакторе Protégé (рис.4.1).

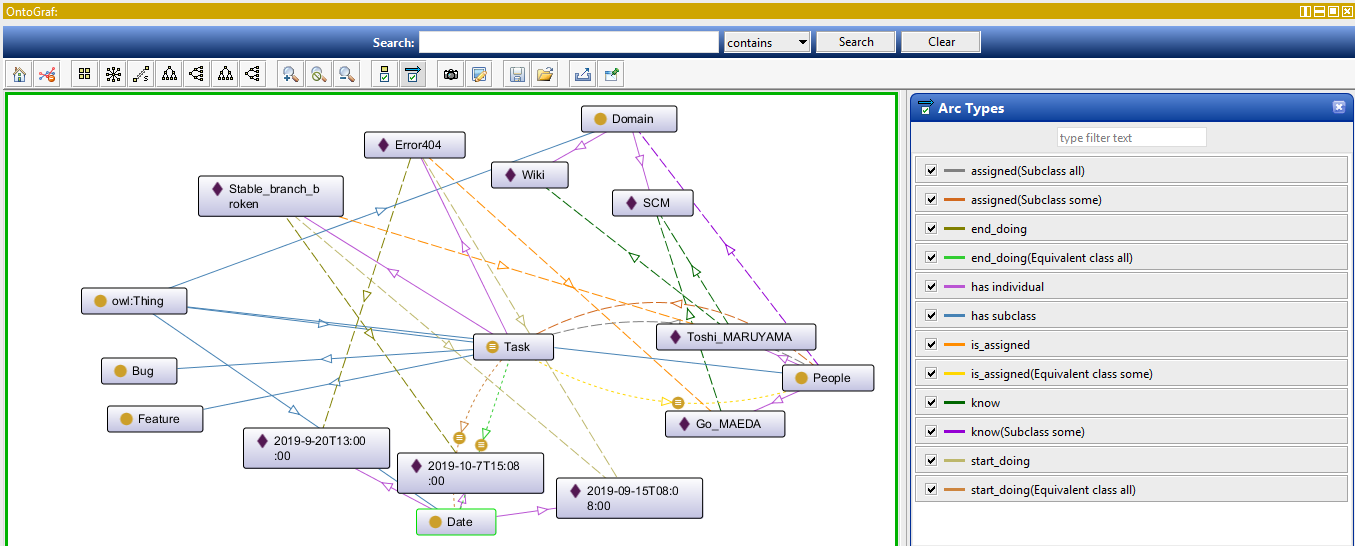


Рис.4.1. Онтология задач с указанием связей классов и индивидуумов

Онтология содержит классы с ограничениями кардинальности, что шире понятий домена и ранга (рис.4.2). Класс Задач (Task) имеет связи назначения (is assigned) Человеку (People), специализации (specialized) по Области применения (Domain), дату создания (start\_doing) по Дате (Date).

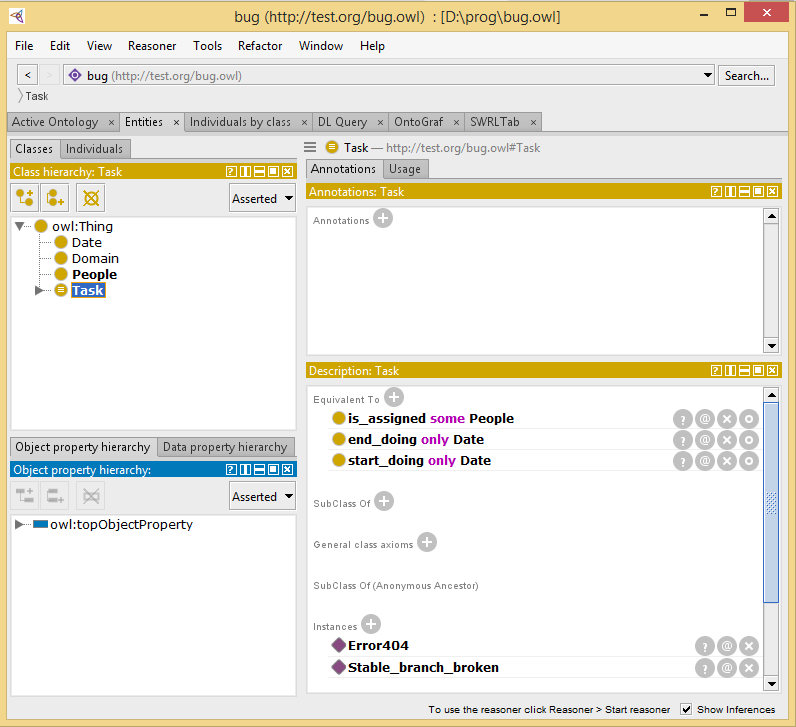


Рис.4.2 Описание класса Задач в онтологии

Библиотека owlready [16] используется для работы с онтологией. Имеется официальное руководство обо всех ее возможностях с примером основных реализаций. Изначально создание и проектирование онтологии осуществлялось программным способом с помощью данной библиотеки. На рис.4.3 отображена загрузка онтологии и вывод классов с их индивидуумами. Для каждой задач также отображены связи соответственно.

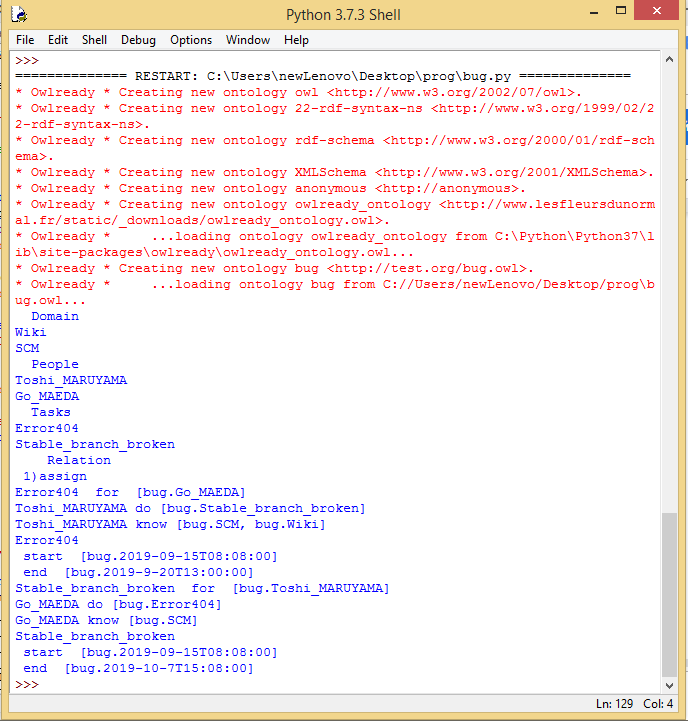


Рис.4.3 Консольное приложение вывода онтологии

Библиотека sys предоставляет доступ к переменным и функциям интерпретатора python. В программе используется для работы с файлами.

Библиотека types интерпретирует стандартные типы Python, а именно для создания нового объекта. А datetime работает с форматом даты и время.

Модуль PyInstaller интерпретирует код программы в исполняемый формат под любую ОС. Следовательно, обеспечивает кросс-платформенность.

## Реализация ПО

В соответствии с требованиями разработан многооконный интерфейс на языке python. Основные функции в интерфейсе:

* Создание задачи с параметрами;
* Распределение созданной задачи по исполнителям.

Для работы программы необходимо запустить файл с соответствующим ОС расширением. Пример на основе Windows для \*.exe файла.

Если у ПК нет доступа к интернету, следует нажать в верхнем меню Import файла онтологии формата owl (рис.5.1). Иначе загружается автоматически по ссылке URI, имеющейся у каждой онтологии.

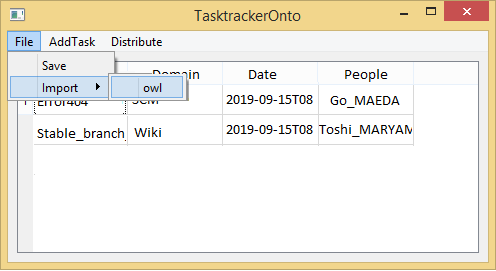


Рис.5.1 Импорт файла онтологии задач

Для добавления новой задачи нужно нажать на соответствующий пункт меню AddTask. Появится новое поле в таблице с возможностью редактирования. Как только значение установлено, активируется запрет на дальнейшие изменения.

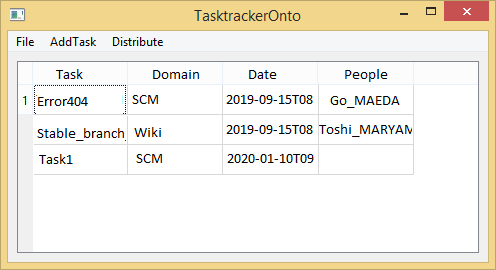


Рис.5.2 Результат работы операции AddTask

Выполнить основную задачу программы и проекта, а именно распределение новой задачи по исполнителям, возможно по нажатию пункта Distribute. Если у каких-то задач нет исполнителя, то можно выбрать данную задачу и повторить предыдущее действие. Результатом данной операции будет обновление таблицы с назначенным исполнителем (рис.5.3).

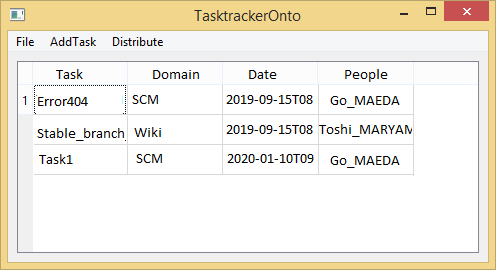


Рис.5.3 Результат работы операции Distribute

Новые данные важно сохранить через пункт меню File > Save (рис.5.1), чтобы при следующей загрузке онтология сохранила новые значения и связи.

# Заключение

Разработанная на данный момент система частично сопоставляет активные задачи исполнителям с применением онтологии и прецедентного подхода.

Для этого изучены материалы по онтологии в бизнес-процессах по управлению задачами, рассмотрены аналоги решения распределения задач как с применением онтологии, так и другими инструментами интеллектуального анализа. Выбранная среда разработки python IDLE и множество поддерживаемых библиотек, ориентированных на интеллектуальный анализ, помогли реализовать desktop-приложение распределения задачи по группам исполнителей.

Дополнительная формализация параметров задачи позволила управлять онтологией. Ограничения онтологии ПрО через связи и кардинальности сузили набор для распределения. Реализованные функции включают в себя простой старт и удобный интерфейс, визуализацию онтологии IT-задач в виде таблицы, отмеченных в аналогах.

Для анализа качества работы распределения достаточно использовать дедукцию, а именно построить небольшую часть системы в онтологии, как это было сделано из ресурсов Redmine, и рассчитывать качество с изменением небольшого числа параметров.

При дальнейшем исследовании планируется экспериментировать методами классификации для выбора оптимального алгоритма. Так же будет использоваться менее формализованный язык, чем OWL. В высоко-формализованном представлении не оказалось смысла. Однако необходим так же повторный выбор инструментов для интеграции онтологии с программой.

# Библиографический список

1. Мошкин В.С. Онтологический подход к анализу временных рядов / Н.Г. Ярушкина, В.С.Мошкин// Интегрированные модели и мягкие вычисления в искусственном интеллекте. Сб. научных трудов VII-й Международной научно-практической конференции. – Коломна – 2013. – № 2. – С. 529–537
2. Мошкин В.С. Онтология как основа семантического анализа данных в задачах автоматизированного проектирования / Н.Г. Ярушкина, В.С.Мошкин// Нечеткие системы и мягкие вычисления (НСМВ-2014): труды Шестой всероссийской научно – практической конференции. – Т.1. –СПб. : Политехникасервис, 2014. – С. 174-182.
3. Карпов Л.Е. Методы добычи данных при построении локальной метрики в системах вывода по прецедентам /Л. Е. Карпов, В. Н. Юдин//Препринты Института системного программирования РАН, Препринт 18, 2006 г. <http://citforum.ru/consulting/BI/data_mining/>
4. Правила принятия решений на основе онтологии задач с помощью правил SWRL <https://cyberleninka.ru/article/n/upravlenie-prinyatiem-resheniy-v-organizatsionnom-upravlenii-s-primeneniem-pravil/viewer>
5. Никулина Н. О., Черняховская Л. Р. Модели информационной поддержки принятия решений с использованием систем управления электронным документооборотом: учеб. пособие. Уфа: УГАТУ, 2009. 134с.
6. Автоматическое назначение задач в Jira с помощью ML. – 24.06.2018. [Электронный доступ] <https://m.habr.com/ru/company/hh/blog/457418/>
7. Создание семантического веб-приложения. – Учебный проект. – 2011 [Электронный доступ] <https://habr.com/ru/post/123612/>
8. Tiller: тайм-менеджмент оффлайн. //Блог о гаджетах и технологиях / Гаджеты / InspectorGadgets.ru, 17.04.2018 [Электронный доступ] <https://www.inspectorgadgets.ru/post/tiller-time-tracker-kickstarter>
9. Устройство трекер Tiller для учета времени, потраченного на каждую задачу.
10. Загорулько Ю.А., Онтологии и их практическое применение в системах, основанных на знаниях/ Ю.А. Загорулько, Г.Б. Загорулько// Цифровой репозиторий DSpace ИВТ СО РАН

1. [Лапшин](https://rsdn.org/article/philosophy/what-is-onto.xml) В.А., Роль онтологий в современной компьютерной науке / В.А. Лапшин. – RSDN Magazine #4. – 2009
2. Онтология для управления проектами: обследование. / International Journal of Information Processing and Management. – ноябрь 2014 [Электронный доступ] <https://www.researchgate.net/publication/270281571_Ontologies_for_Project_Management_Survey/download>
3. ВласовД.Ю. Автоматизация извлечения отношений между понятиями из текстов естественного языка /Д.Ю. Власов,Д.Е. Пальчунов, П.А. Степанов // Вестник Новосибирского государственного университета. Серия: Информационные технологии. – 2010 [Электронный доступ] https://cyberleninka.ru/article/n/avtomatizatsiya-izvlecheniya-otnosheniy-mezhdu-ponyatiyami-iz-tekstov-estestvennogo-yazyka
4. ЛеХоай. Семантическое аннотирование документов в электронных библиотеках/ ЛеХоай, А.Ф. Тузовский // Известия Томского политехнического университета. Инжиниринг георесурсов. – 2013 [Электронный доступ] https://cyberleninka.ru/article/n/semanticheskoe-annotirovanie-dokumentov-v-elektronnyh-bibliotekah
5. Абрамян Т.С., Анисимов А.С., Коровин М.Д. Разработка семантической модели подбора исполнителей на основании его компетенции / Т.С. Абрамян, А.С. Анисимов, М.Д. Коровин // Intelligenttechnologiesforinformationprocessingandmanagement (itipm'2015). -2015. - С.29-32
6. Клименко С.В., Золотарев О.В., Шарнин М.М. Использование онтологического подхода для анализа текстов естественного языка / С.В. Клименко, О.В. Золотарев, М.М. Шарнин // Вестник Российского нового университета. -2017. -№1. – С. 67-71
7. Святогор Л., Гладун В. Семантический анализ текстов естественного языка: цели и средства. – 2009. - Knowledge-Dialogue-Solution
8. Jean-Philippe Lang. Сайт Redmine. Developer’s – 2006-2014 [Электронный доступ] https://www.redmine.org/projects/redmine/issues/report
9. Тушканова О.Н., Городецкий В.И. Ассоциативная классификация: аналитический обзор. Часть 1 // Труды СПИИРАН. 2015. № 38 (1). C. 183-203. [Электронный доступ] https://doi.org/10.15622/sp.38.10

1. Актёр – элемент, обозначающий роли пользователя, который взаимодействует с определенной сущностью [↑](#footnote-ref-1)