**АННОТАЦИЯ**

В данном программном документе приведена пояснительная записка к программе «task\_manager-1.0» («Плагин для платформы IntelliJ для мониторинга процесса создания программы и формирования отчета»), предназначенной для создания и управления задачами проекта и генерации отчета о проделанной работе.

В разделе «Введение» указано наименование программы, краткое наименование программы и документы, на основании которых ведется разработка.

В разделе «Назначение и область применения» указано функциональное назначение программы, эксплуатационное назначение программы и краткая характеристика области применения программы.

В разделе «Технические характеристики» содержатся следующие подразделы:

* постановка задачи на разработку программы;
* описание алгоритма и функционирования программы с обоснованием выбора схемы алгоритма решения задачи и возможные взаимодействия программы с другими программами;
* описание и обоснование выбора метода организации входных и выходных данных;
* описание и обоснование выбора состава технических и программных средств.

В разделе «Ожидаемые технико-экономические показатели» указана предполагаемая потребность и экономические преимущества разработки по сравнению с отечественными и зарубежными образцами или аналогами

Настоящий документ разработан в соответствии с требованиями:

1. ГОСТ 19.101-77 Виды программ и программных документов [1];
2. ГОСТ 19.102-77 Стадии разработки [2];
3. ГОСТ 19.103-77 Обозначения программ и программных документов [3];
4. ГОСТ 19.104-78 Основные надписи [4];
5. ГОСТ 19.105-78 Общие требования к программным документам [5];
6. ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом [6];
7. ГОСТ 19.404-79 Пояснительная записка. Требования к содержанию и оформлению [7].

Изменения к Пояснительной записке оформляются согласно ГОСТ 19.603-78 [8], ГОСТ 19.604-78 [9].

Перед прочтением данного документа рекомендуется ознакомиться с терминологией, приведенной в Приложении 1 настоящей пояснительной записки.

**СОДЕРЖАНИЕ**

[1. ВВЕДЕНИЕ 2](#_Toc430461508)

[1.1. Наименование программы 2](#_Toc430461509)

[1.2. Условное обозначение темы разработки 2](#_Toc430461510)

[1.3. Документы, на основании которых ведется разработка 2](#_Toc430461511)

[2. НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ 3](#_Toc430461512)

[2.1. Назначение программы 3](#_Toc430461513)

[2.1.1. Функциональное назначение 3](#_Toc430461514)

[2.1.2. Эксплуатационное назначение 3](#_Toc430461515)

[2.2. Область применения программы 3](#_Toc430461516)

[3. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ 4](#_Toc430461517)

[3.1. Постановка задачи на разработку программы 4](#_Toc430461518)

[3.2. Описание алгоритма и функционирования программы 4](#_Toc430461519)

[3.2.1. Описание алгоритма программы 4](#_Toc430461520)

[3.2.1.1. Создание профиля зуба эвольвентного соединения 4](#_Toc430461521)

[3.2.1.2. Создание цилиндрических прямозубых зубчатых колес 5](#_Toc430461522)

[3.2.1.3. Создание конических прямозубых зубчатых колес 5](#_Toc430461523)

[3.2.1.4. Создание винтовых криволинейных зубчатых колес 6](#_Toc430461524)

[3.2.1.5. Алгоритм определения наличия пересечения цилиндров 7](#_Toc430461525)

[3.2.2. Обоснование выбора алгоритма решения задачи 7](#_Toc430461526)

[3.2.3. Возможные взаимодействия программы с другими программами 8](#_Toc430461527)

[3.3. Описание и обоснование выбора метода организации входных и выходных данных 8](#_Toc430461528)

[3.4. Описание и обоснование выбора состава технических и программных средств 8](#_Toc430461529)

[4. ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ 9](#_Toc430461530)

[4.1. Предполагаемая потребность 9](#_Toc430461531)

[4.2. Экономические преимущества разработки по сравнению с отечественными и зарубежными аналогами 9](#_Toc430461532)

[5. ИСТОЧНИКИ, ИСПОЛЬЗОВАННЫЕ ПРИ РАЗРАБОТКЕ 10](#_Toc430461533)

[ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ ОСНОВНЫХ МОДУЛЕЙ 11](#_Toc430461534)

[ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ 12](#_Toc430461535)

# ВВЕДЕНИЕ

## Наименование программы

Наименование программы – «Плагин для платформы IntelliJ для мониторинга процесса создания программы и формирования отчета».

Наименование программы на английском языке – «Plugin for IntelliJ Platform for monitoring of developing process and report generation».

## Документы, на основании которых ведется разработка

Разработка ведется на основании приказа Национального исследовательского университета «Высшая школа экономики» № \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ от \_\_\_\_\_\_\_\_\_\_ «наименование приказа»

# НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ

## Назначение программы

### Функциональное назначение

Функциональным назначением программы является мониторинг процесса разработки программного продукта разработчиком/командой разработчиков и создание отчета на основе полученных данных. Плагин собирает информацию, которая предоставляется платформой для разработки программ IntelliJ, сохраняет соответствующие данные и формирует отчеты за установленный период времени в виде .CSV файлов. С помощью данной программы можно планировать дедлайны по задачам, планировать разработку проекта.

### Эксплуатационное назначение

Предполагается использование программного интерфейса при разработке ПО непосредственно заказчиками для анализа продуктивности выполнения заказа исполнителем. Программа будет полезна в случае создания фриланс-сервиса для мониторинга заказчиками процесса разработки продукта. Также возможно использование плагина в командных проектах менеджерами проектов. Основным преимуществом данной программы является её непосредственное внедрение в инфраструктуру IntelliJ, что позволяет быстро управлять задачами и создавать отчеты, не отвлекаясь от разработки быстро управлять задачами и создавать отчеты.

## Область применения программы

Программа носит прикладной характер. Программа подразумевает запись и сохранение отчетности о проделанной работе программистом (исполнителем) для последующего предоставления данных о процессе разработки продукта заказчику. Программа использует данные, предоставляемые платформой IntelliJ для мониторинга информации о процессе разработки и создания соответствующих записей о проделанной работе.

# ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

## Постановка задачи на разработку программы

Постановка задачи – разработка плагина для открытой платформы разработки ПО IntelliJ, который позволить управлять задачами разработки проектов, автоматически отсчитывать и запоминать время, отведенное под реализацию задач, автоматически формировать отчеты по проделанной работе за указанный период. Плагин также должен представлять собой платформу для дальнейшего расширения функциональности последнего.

## Описание алгоритма и функционирования программы

### Описание алгоритма программы

Так как сложность программы по большей части составляют не алгоритмы, а непосредственно сама архитектура, то далее ей будет уделено большее внимание. Программа состоит из трех пакетов-компонент:

* базового модуля, содержащего классы и методы для работы бизнес-логики приложения;
* модуля, содержащего классы и методы, формирующие пользовательский интерфейс программы и взаимодействующие с бизнес-моделью;
* модуля, содержащего классы и методы, взаимодействующие с платформой IDE IntelliJ и обрабатывающие определенную информацию о текущем состоянии проекта.

### Регистрация объектов-менеджеров и событий в IDE

Для корректной работы плагина все необходимые менеджеры-синглтоны, а также события регистрируются в файле plugin.xml, в котором группируются соответственно архитектуре самого плагина. Так как большая часть архитектуры самой платформы IDE построена на регистрации и вызовах событий, а также на создании объектов-сервисов (объектов-менеджеров), то и данный плагин не стал исключением (далее будут рассмотрены все менеджеры данного плагина).

### Алгоритмы и классы базового модуля

Базовый модуль (*org.taimuraztibilov.taskmanager.base*) содержит в себе классы, представляющие основную бизнес-модель приложения. Классы *Project, Milestone, Task, KeyPoint, Label* представляют собой модели базы данных SQLite, ER-диаграмма которой представлена ниже (Рисунок 1).

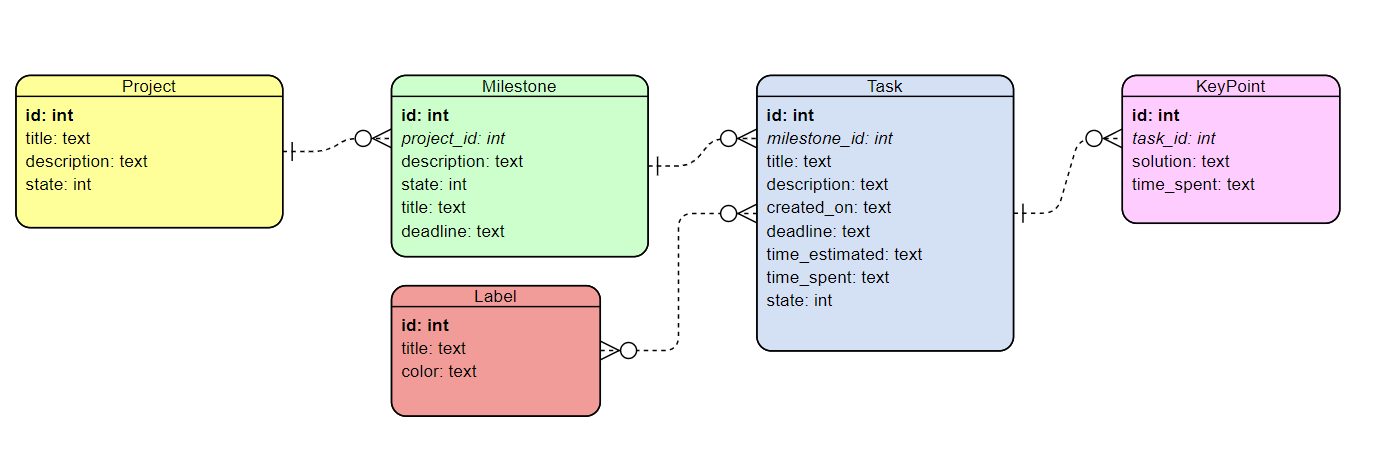
Все сущности и поля базы данных реализованы по аналогии GitLab (в том числе и для последующего расширения в сторону последнего), который на уровне ведения репозитория предоставляет похожий функционал.

Рисунок . ER-диаграмма базы данных

Для реализации функциональной части плагина используются четыре синглтона-менеджера:

1. *DataBaseManager.* Его основная задача – взаимодействие с базой данных плагина. Он также реализует интерфейс *DataEditor*, который позволяет классам-моделям оповещать об изменении значений в своих полях. С помощью JDBC SQLite3 создается подключение к базе данных плагина, в случае необходимости создаются недостающие таблицы. Также в данном классе существуют все необходимые методы для создания, изменения, получения и удаления соответствующих данных из БД.
2. *TimeManager.* Основная задача данного синглтона – это отслеживание времени выполнения выбранной задачи, а также для создания ключевых точек по задаче с описанием проделанной работы за один сеанс разработки в IDE. Далее данные, полученные с его помощью, будут доступны для формирования отчетов.
3. *ReportManager.* Этот синглтон выполняет функцию автоматического формирования отчета о проделанной работе за заданный период времени. Он обращается к базе данных с join-SQL запросом, в котором собирает все ключевые точки задач, закрытых в данном периоде, и с помощью открытой библиотеки openCSV превращает полученные данные в .csv файл, который будет записан в корневой директории текущего проекта, открытого в IDE.
4. *PluginManagerService.* Данный менеджер оперирует основными данными, с которыми в данный момент пользователь взаимодействует через IDE. Он по сути дела является вспомогательным синглтоном, содержащем по большей части метаданные.

Все четыре вышеперечисленных класса регистрируются в платформе IntelliJ как сервисы платформы – это позволено сделать синглтонам, чтобы они подгружались класслоудером самой платформы при открытии программы IDE, и существовали на протяжении всего жизненного цикла программы вплоть до завершения работы в ней. В остальных же случаях классы и объекты будут подгружаться только после происхождения какого-либо из отмеченных событий и будут существовать только пока они определены платформой как используемые.

### Алгоритмы и классы модуля UI

Модуль пользовательского интерфейса (*org.taimuraztibilov. taskmanager.ui*) содержит в себе статические методы, которые позволяют взаимодействовать пользователю с методами, доступными в базовом модуле. На примерах работы и взаимодействия с данными формами можно подробно рассмотреть функционал, доступный в методах базового класса.

Для начала стоит пояснить, что в случае создания окон с пользовательским интерфейсом не все так однозначно, как может показаться на первый взгляд. Дело в том, что базовый фреймворк для создания форм графического интерфейса *Swing* является очень гибким, но и в то же время легковесным, что накладывает некоторые ограничения на создания пользовательских элементов интерфейса.

Здесь для создания последних были использованы конструкторы компонентов, используемых в самой IDE, что являются wrapper классами над стандартными компонентами, что в свою очередь позволяет более детально построить интерфейс и довесить дополнительных элементов. Подробное описание компонентов JB, соответствующих компонентам из базового фреймворка, можно найти на странице документации по дизайн-коду IDE.

При работе с задачами их необходимо сначала создать. Для постройки пользовательских форм используется класс *AddDataFormBuilder,* который в свою очередь содержит соответствующие статические методы, формирующие окна для создания новых записей в БД плагина. Для этого полученные данные из полей интерпретируются в нужном формате, а затем передаются на вход add-методам инстанса *DataBaseManager.*

Для выбора задачи, которую пользователь хочет отслеживать, в статических методах класса *ShowDataFormBuilder* строится 3 разных формы – для выбора проекта, вехи и непосредственно самой задачи. После каждого выбора в полях текущего выбора в *PluginManagerService* запоминаются id элементов, выбранных для отслеживания. Если до этого пользователь отслеживал другую задачу, программа предложит ему зафиксировать точку в разработке. При этом время, затраченное на задачу, *автоматически* подтянется к другим данным, так как *TimeManager* запоминает последнее состояние времени отслеживаемой задачи.

Также при двойном щелчке мышью на элемент в списке просмотра есть возможность просмотреть информацию по выбранному элементу.

Изменение и удаление элементов происходит в окнах и формах, которые строятся в классе *EditDataFormBuilder*. Построенные им формы напоминают формы, строящиеся в классе *AddDataFormBuilder*. Там же и находится метод, который строит форму для создания отчета, в которой пользователь должен указать свои ФИО, а также организацию, если таковая имеет место быть.

### Алгоритмы и классы модуля Action

Архитектура всех без исключения плагинов для IntelliJ основывается на реализации событий и групп событий. В модуле событий (org.taimuraztibilov.taskmanager.action) содержаться все реализации событий плагина. Все они имеют предикат, при выполнении которого они становятся доступными для пользователя. Для удобства их использования они были сгруппированы по функциональному признаку (например, группа Add). Все события описываются в конфигурационном файле плагина *plugin.xml* (пример описания можно рассмотреть на Рисунке 2).

Ниже представлено сопоставление класса события с его описанием:

### Создание винтовых криволинейных зубчатых колес

На вход алгоритму даются – количество зубцов у колеса,

– нормальный угол зацепления, – угол наклона линии зуба, – ширина колеса,

– нормальный модуль, -радиус вала.

Вначале высчитываются все необходимые значения

* – диаметр длительной окружности (pitch circle diameter)
* – окружной угол зацепления (radial pressure angle)
* - радиус основной окружности (base circle radius)
* - радиус окружности вершин зубьев (outside circle radius)
* - радиус окружности впадин зубьев (root circle radius)

Далее строится колесо без зубьев, имеющее внутренний радиус равный , внешний равный и ширину равную .

После этого строятся зубья. Зуб представляют собой движущийся по винтовой линии профиль зуба, описанный в 3.2.1.1 . Винтовая линия имеет следующие параметрические уравнения в цилиндрической системе координат.

имеет диапазон

Эти зубья расположены на расстоянии радиан друг от друга.

### Алгоритм определения наличия пересечения цилиндров

Для шестеренок и осей хорошо подходит цилиндр, как ограничивающая фигура. Поэтому для определения пересечения шестеренок и осей используется алгоритм пересечения цилиндров.

На вход алгоритму даются –центры цилиндров, – вектора направления осей цилиндров, – радиусы цилиндров, – высоты цилиндров.

Два цилиндра не пересекаются, если существует прямая, на которой их проекции не пересекаются. Пусть такая прямая называется разделяющей прямой. Так как параллельный перенос прямой разделяющей прямой является разделяющей прямой, то можно провести прямую через начало координат. Такая прямая будет определяться разделяющим вектором .

Пусть и . Тогда тест на пересечение выглядит следующим образом

Вначале проверим случаи, когда направление разделяющего вектора совпадет с осями: и .

Если прошлая проверка не дала положительных результатов, проверим случаи, когда перпендикулярно осям. Такую проверку можно сделать на полукруге, поскольку разделяющий вектор и эквивалентны. Поиск нужного вектора сводится к поиску корня выпуклой функции. Для этого хорошо подходит метод бисекции.

Если разделяющий вектор до сих пор не был найден, то проводится поиск по всем остальным направлениям. Эта задача сводится к поиску минимума на выпуклой двумерной функции с линейными ограничениями. Эта задача может быть решена методом внутренней точки.

Подробное описание алгоритма описано в [2].

### Обоснование выбора алгоритма решения задачи

Алгоритмы 1-4 являются алгоритмами построения по формулам, описывающими зубчатые колеса, для них нет точных алгоритмов построения поэтому они были написаны самостоятельно с учетом механики фреймворка.

Алгоритмов для определения пересечения цилиндров не так много (мною было найдено 3) и все они отличаются сложным математическим аппаратом. В связи с этим был выбран тот, который использует более простой математический аппарат. Это позволило написать код, который легко отлаживать и поддерживать.

### Возможные взаимодействия программы с другими программами

Программа является standalone web приложением и не предполагает взаимодействия ни с какими другими программами, кроме браузера в которой она запускается.

## Описание и обоснование выбора метода организации входных и выходных данных

Входные данные могут задаваться двумя способами:

* Данные могут задаваться пользователю через специальную форму. Эта форма показывается пользователю при создании ведущей оси, добавлении любой шестерёнки на ось и к другим шестеренкам.
* Данные могут загружаться в виде JSON файла, который был ранее сохранен приложением

Выходными данными является JSON файл в котором сохраняется состояние зубчатой передачи. Этот файл можно потом загрузить для восстановления сохраненной зубчатой передачи.

## Описание и обоснование выбора состава технических и программных средств

Для корректной работы программы требуются следующие технические и программные средства:

1. процессор с частотой 1ГГц или более;
2. 1 гигабайт (ГБ) (для 32-разрядной системы) или 2 ГБ (для 64-разрядной системы) или более;
3. не менее 2 ГБ места на жестком диске;
4. монитор и видеоадаптер с разрешением 800 X 600;
5. мышь или совместимое указывающее устройство и клавиатура;
6. операционная система Microsoft Windows 7 или выше, Linux, Mac OS;
7. Браузер с поддержкой HTML5 и WebGL. Для сохранения и загрузки фалов требуется поддержка Map из спецификации ES6.

# ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ

## Предполагаемая потребность

Программа может потребоваться в университетах, профессионально-технических училищах, техникумах и.т.д в качестве демонстрации работы зубчатых передач. Так как программа является бесплатной и может быть распространена по ссылке, то ее передача ученикам и показ на лекциях крайне удобны

## Экономические преимущества разработки по сравнению с отечественными и зарубежными аналогами

На момент создания приложения в сети не было обнаружено других приложений, позволяющих моделировать 3D зубчатые передачи в вебе.

Среди аналогов можно geargenerator.com который позволяет моделировать в вебе 2D цилиндрические прямозубые передачи и различные САПР для которых уже были реализованы различные зубчатые передачи.

# ИСТОЧНИКИ, ИСПОЛЬЗОВАННЫЕ ПРИ РАЗРАБОТКЕ

1. SDP/SI «Elements of Metric Gear Technology », 2010
2. David Eberly «Intersection of Cylinders» 2000-2015
3. Jeffery Baker, Jason Reynolds, Stephen Tecce «Parametric Design of a Spiral Gear Process», 2011
4. GEARS Educational Systems «Spur Gear Terms and Concepts»,2004
5. Gear Trains [Электронный ресурс] //URL: http://moodlearn.ariel.ac.il/pluginfile.php/837913/mod\_resource/content/0/%D7%9E%D7%9E%D7%A1%D7%A8%D7%95%D7%AA.pdf

Оформление не по ГОСТ

**ПРИЛОЖЕНИЕ 1**

# ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ ОСНОВНЫХ МОДУЛЕЙ

|  |  |
| --- | --- |
| **Модуль** | **Назначение** |
| *app* | Входная точка программы. Инициализирует элементы UI и контроллеры. Выставляет настройки. |
| *сontrollers/GearTrainController* | Контроллер, обеспечивающий связь между пользователем и моделью зубчатой передачи |
| *controllers/MouseController* | Контроллер, обеспечивающий связь между мышкой и моделями зубчатых колес и валов |
| *math/Cylinder*  Нет описания классов (полей и методов),  Нет диаграммы классов | Представляет собой цилиндр как математическую фигуру. Используется для определения пересечений цилиндров. |
| *models/BaseRotatingPart* | Представляет собой базовый класс для всех моделей системы, совершающих вращение |
| *models/Shaft* | Представляет модель вала |
| *models/Gear* | Представляет собой базовый класс для всех моделей зубчатых колес |
| *models/SpurGear* | Представляет собой модель цилиндрического прямозубого зубчатого колеса |
| *models/BevelGear* | Представляет собой модель конического прямозубого зубчатого колеса |
| *models/HelicalGear* | Представляет собой модель винтового криволинейного зубчатого колеса |
| *models/GearTrain* | Представляет собой модель зубчатой передачи. |
| *ui/InfoBox* | Класс отвечающий за управление элементом, отвечающим за вывод информации о моделях |
| *ui/MesssageBox* | Класс отвечающий за управление элементом, отвечающим за вывод сообщений и ошибок. |
| *ui/ModalForm* | Класс создающий и отвечающий за форму ввода параметров |
| *views/BasicRotatingPartMesh* | Базовый класс mesh представляющий все 3D представления моделей |
| *views/ShaftMesh* | Класс mesh представляющий 3D модель вала |
| *views/SpurGearMesh* | Класс mesh представляющий 3D модель цилиндрического прямозубого зубчатого колеса |
| *views/BevelGearMesh* | Класс mesh представляющий 3D модель конического прямозубого зубчатого колеса |
| *views/HelicalGearMesh* | Класс mesh представляющий 3D модель винтового криволинейного зубчатого колеса |

# ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Лист регистрации изменений | | | | | | | | | |
| Номера листов (страниц) | | | | | Всего листов (страниц в докум.) | № документа | Входящий № сопрово-дительно-  го докум. и дата | Подл. | Да-  та |
| Изм. | Изменен-  ных | Заменен-  ных | новых | аннулированных |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  | Не обязательно |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |