**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ**

**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИ УНИВЕРСИТЕТ**

**«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук

Департамент программной инженерии

|  |  |
| --- | --- |
| **Подп. и дата** |  |
| **Инв № дубл** |  |
| **Взам. инв. №** |  |
| **Подп. и дата** |  |
| **Инв. № подл** |  |

|  |  |  |
| --- | --- | --- |
| СОГЛАСОВАНО  Профессор,  ведущий научный сотрудник научно-учебной лаборатории ПОИС факультета компьютерных наук  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.А. Каленкова  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2019 г. |  | УТВЕРЖДАЮ  Академический руководитель образовательной программы «Программная инженерия»  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.В. Шилов  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2019 г. |

**Программа анализа логов событий систем отслеживания ошибок**

**Пояснительная записка**

**ЛИСТ УТВЕРЖДЕНИЯ**

**RU.17701729.04.13-01 81 01-1-ЛУ**

**Исполнитель**

Студент группы БПИ173

\_\_\_\_\_\_\_\_\_\_\_\_ / С. И. Ройтман /

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2019 г.

**УТВЕРЖДЕНО**

**RU.17701729.04.13-01 81 01-1-ЛУ**

**Программа анализа логов событий систем отслеживания ошибок**

**Пояснительная записка**

**RU.17701729.04.13-01 81 01-1-ЛУ**

**Листов 25**

|  |  |
| --- | --- |
| **Подп. и дата** |  |
| **Инв № дубл** |  |
| **Взам. инв. №** |  |
| **Подп. и дата** |  |
| **Инв. № подл** |  |

**ОГЛАВЛЕНИЕ**

[1. ВВЕДЕНИЕ 5](#_Toc8658306)

[1.1. Наименование программы 5](#_Toc8658307)

[1.2. Краткая характеристика области применения программы 5](#_Toc8658308)

[2. ОСНОВАНИЯ ДЛЯ РАЗРАБОТКИ 5](#_Toc8658309)

[3. НАЗНАЧЕНИЕ РАЗРАБОТКИ 6](#_Toc8658310)

[3.1. Функциональное назначение 6](#_Toc8658311)

[3.2. Эксплуатационное назначение 6](#_Toc8658312)

[4. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ 7](#_Toc8658313)

[4.1. Постановка задачи на разработку программы 7](#_Toc8658314)

[4.2. Описание алгоритма и функционирования программы 7](#_Toc8658315)

[4.2.1. Алгоритм кластеризации 7](#_Toc8658316)

[4.2.2. Алгоритмы расчета расстояния между объектами 8](#_Toc8658317)

[4.2.2.1. Расстояние Дамерау - Левенштейна 8](#_Toc8658318)

[4.2.2.2. Расстояние через модели кластеризации 11](#_Toc8658319)

[4.2.3. Результаты кластеризации 11](#_Toc8658320)

[4.3. Обоснование выбора алгоритма решения задачи 12](#_Toc8658321)

[4.4. Описание и обоснование выбора метода организации входных и выходных данных 12](#_Toc8658322)

[4.5. Описание и обоснование выбора состава технических и программных средств 12](#_Toc8658323)

[4.5.1. Описание выбора состава технических средств 12](#_Toc8658324)

[4.5.2. Обоснование выбора состава технических средств 13](#_Toc8658325)

[4.5.3. Описание выбора состава программных средств 13](#_Toc8658326)

[4.5.4. Обоснование выбора состава программных средств 13](#_Toc8658327)

[5. ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ 14](#_Toc8658328)

[5.1. Предполагаемая потребность 14](#_Toc8658329)

[5.2. Ориентировочная экономическая 14](#_Toc8658330)

[5.3. Экономические преимущества разработки по сравнению с отечественными и зарубежными аналогами 14](#_Toc8658331)

[6. ИСТОЧНИКИ, ИСПОЛЬЗОВАННЫЕ ПРИ РАЗРАБОТКЕ 15](#_Toc8658332)

[ПРИЛОЖЕНИЕ 1 16](#_Toc8658333)

[ПРИЛОЖЕНИЕ 2 17](#_Toc8658334)

[6.2. Описание и функциональное назначение классов 17](#_Toc8658335)

[6.3. Описание и функциональное назначение полей и методов классов 18](#_Toc8658336)

[ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ 25](#_Toc8658337)

**АННОТАЦИЯ**

В данном программном документе приведена пояснительная записка к «Программе анализа логов событий систем отслеживания ошибок», предназначенной для кластеризации исполнителей событий логов систем отслеживания ошибок.

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.

# ВВЕДЕНИЕ

## Наименование программы

Наименование программы: «Программа анализа логов событий систем отслеживания ошибок» («A Program for the Analysis of Event Logs of Bug Tracking Systems»).

## Краткая характеристика области применения программы

Полученные методы могут быть применены в индустрии для глубинного анализа процессов (см. терминологию в Приложении 1) тестовых систем.

# ОСНОВАНИЯ ДЛЯ РАЗРАБОТКИ

Программа выполнена в рамках темы курсовой работы — «Программа анализа логов событий систем отслеживания ошибок», в соответствии с учебным планом подготовки бакалавров по направлению 09.03.04 «Программная инженерия».

Основанием для разработки приказ декана ФКН И.В. Аржанцева № 2.3-02/1012-01 от 10.12.18

**Наименование темы разработки:** «Программа анализа логов событий систем отслеживания ошибок».

**Условное обозначение темы разработки:** «A Program for the Analysis of Event Logs of Bug Tracking Systems».

# НАЗНАЧЕНИЕ РАЗРАБОТКИ

## Функциональное назначение

Функциональным назначением программы является автоматический поиск групп пользователей системы отслеживания ошибок, которые ведут себя схожим образом, например:

* Группа тестировщиков и группа разработчиков
* Группа разработчиков, схожих по количеству решенных проблем

Исходные данные представляют собой загруженные в программу из внешних источников логи событий систем отслеживания ошибок в формате *.xes*, а также конфигурацию работы программы в формате *.csv*.

В качестве выходных данных программа предоставляет визуализацию кластеризации исполнителей процессов, а также файл формата *.csv* с данными о исполнителях (количество событий разных типов, общее количество событий, значение функции) и два файла формата *.dot*, один из которых содержит данные для визуализации кластеров исполнителей, другой данные для визуализации последовательности действий для каждого исполнителя.

## Эксплуатационное назначение

Полученные, в ходе работы над программой, методы могут быть применимы в IT-индустрии для кластеризации исполнителей событий из логов тестирующих систем.

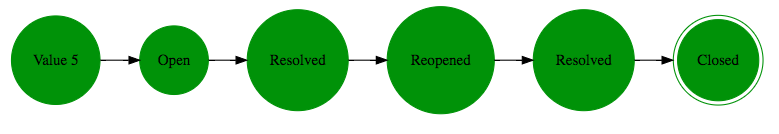
# ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

## Постановка задачи на разработку программы

Необходимо разработать плагин для фреймворка ProM[11]. Разрабатываемый плагин должен:

1. Автоматически производить поиск групп исполнителей событий системы отслеживания ошибок, которые ведут себя схожим образом.
2. Визуализировать найденные группы пользователей.

## Описание алгоритма и функционирования программы

Программой производится кластеризации всех трейсов работ над задачами (см. терминологию в Приложении 1).

*Картинка 1. Трейс исполнителя Value 5 над одной из задач*

Исходя из результатов кластеризации программа вычисляет принадлежность исполнителя к тому или иному кластеру программистов. Например, у исполнителя Value 5 большинство трейсов принадлежат к кластеру низких по эффективности трейсов, это позволяет отнести его к классу неэффективных разработчиков.

* + 1. Алгоритм кластеризации

В качестве метода кластеризации был выбран **метод k-средних (k-means)**, описанный в статье «K-means clustering» [12].

Шаги алгоритма:

1. Определить k – количество кластеров, которое должно быть сформировано
2. Случайным образом выбрать k центроидов (см. терминологию в Приложении 1)
3. Вычислить расстояние от всех объектов до центроидов
4. Для каждого объекта определить ближайший к нему центроид
5. Сместить центроиды кластеров в новые, определив их как центр тяжести всех точек каждого кластера.
6. Вернуться к шагу 3, если новые центроиды не совпадают со старыми

Исходный алгоритм k-средних был модифицирован, а именно шаг

5. Выбор новых центроидов: так как центры тяжести (среднее арифметическое) точек кластера не представляется возможным посчитать, было принято решение определять новый центроид кластера, как объект с наименьшей суммой квадратов расстояний до всех других точек в кластере.

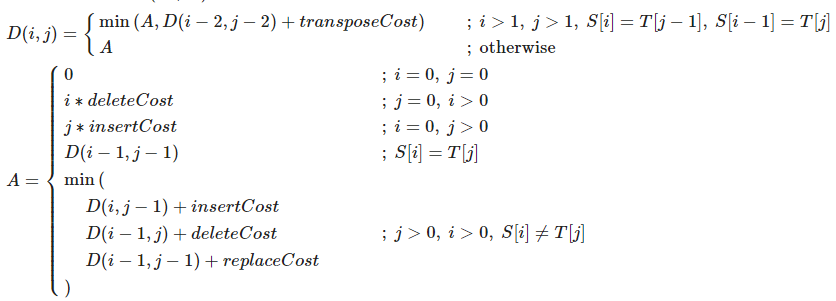
* + 1. Алгоритмы расчета расстояния между объектами
       1. Расстояние Дамерау - Левенштейна

**Расстояние Дамерау – Левенштейна**[13] – это мера разницы двух строк символов, определяемая как минимальное количество операций вставки, удаления, замены и транспозиции (перестановки двух соседних символов), необходимых для перевода одной строки в другую. Является модификацией расстояния Левенштейна: к операциям вставки, удаления и замены символов, определённых в расстоянии Левенштейна добавлена операция транспозиции символов.

В данном случае под «строками» представлены трейсы действий программистов над задачами.

Для расчета расстояния Дамерау – Левенштейна был использован алгоритм   
**Вагнера – Фишера**[14]**:**

Пусть S1 и S2 — две строки (длиной M и N соответственно) над некоторым алфавитом, тогда редакционное расстояние D(S1,S2) можно подсчитать по следующей рекуррентной формуле:

*Картинка 1. Формула подсчета расстояния*

Здесь D(i, j) — расстояние между префиксами строк: первыми i символами строки S и первыми j символами строки T)

Таким образом для получения ответа необходимо заполнить матрицу D, пользуясь рекуррентным соотношением. Ответ соответственно будет храниться в D(M, N).

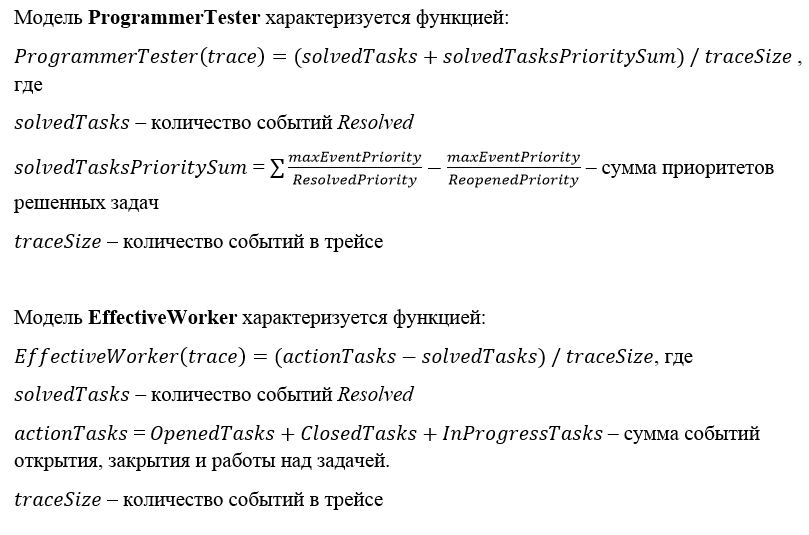
Сложность алгоритма: O(M⋅N)

Затраты памяти: O(M⋅N)

Для анализа кластерного разбиения с помощью расстояния Дамерау-Левенштейна в программе были созданы две модели разбиения:

* тестировщики и разработчики (**ProgrammerTester**)
* разработчики в зависимости от их эффективности (**EffectiveWorker**)

Каждая из моделей характеризуется функцией, которая получает на вход трейс, и выдает значение, характеризующее трейс по заданной модели.

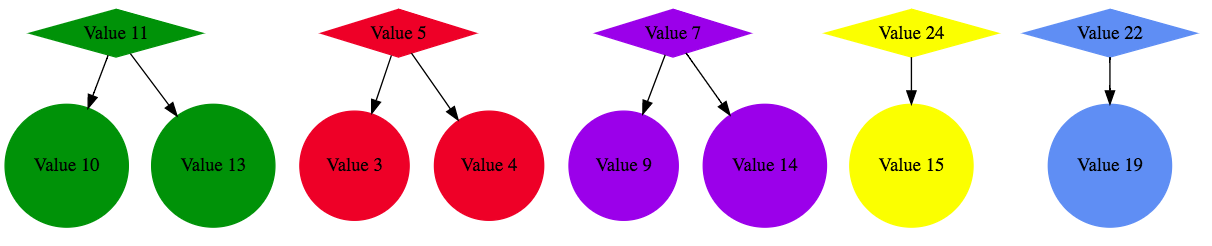


* + - 1. Расстояние через модели кластеризации

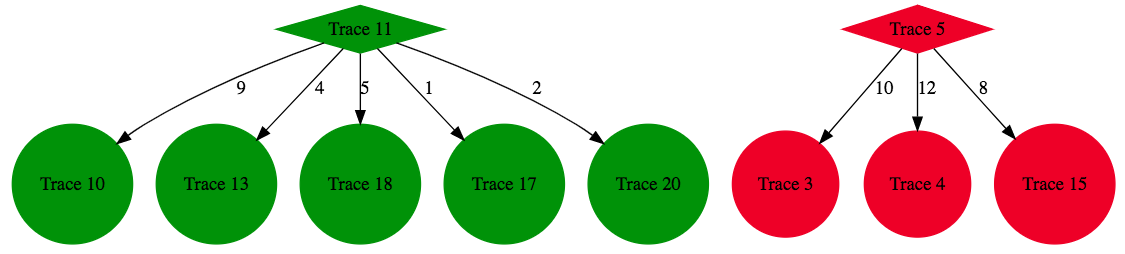
Для анализа кластерного разбиения с использованием расстояния Дамерау – Левенштейна была составлена мера расстояния равная абсолютной разницу функций выбранной модели, т.е.

*ModelBased(traceA, traceB) =* |*F(traceA) – F(traceB)* |

, где F(trace) – функция выбранной модели разделения

* + 1. Результаты кластеризации

*Картинка 2.Один из возможных вариантов разбиения исполнителей по кластерам*



*Картинка 3. Один из возможных вариантов разбиения трейс по кластерам с указанием расстояние до центроидов кластера*

В конкретном примере приведенном сверху с помощью модели EffectiveWorker было выяснено, что зеленый кластер трейсов на 100% состоит из эффективных трейсов. Соответственно исполнители у которых таких трейсов большинство относятся к кластеру эффективных разработчиков.

## Обоснование выбора алгоритма решения задачи

Так как трассы действий исполнителя по задаче являются относительно короткими, например трасса Open->Resolved->Closed, то идея сравнить их между собой через расстояния Дамерау-Левенштейна имеет смысл. Кроме этого идея является новой и не применялась раньше при анализе логов систем.

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

Исходные данные представляют собой загруженные в программу из внешних источников логи событий систем отслеживания ошибок в формате *.xes*, а также конфигурацию работы программы в формате *.csv*. Конфигурация программы представлена в формате *.csv* из-за простоты обработки такого формата, а также отсутствия необходимости часто менять параметры запуска.

В качестве выходных данных программа предоставляет визуализацию кластеризации исполнителей процессов, а также файл формата *.csv* с данными о исполнителях (количество событий разных типов, общее количество событий, значение функции) и два файла формата *.dot*, один из которых содержит данные для визуализации кластеров исполнителей, другой данные для визуализации последовательности действий для каждого исполнителя. Данные о исполнителях представлены в формате *.csv* из-за удобства последующей обработки данных в табличной форме. Визуализация представлена в формате *.dot* для отображения с помощью специальных программ, например Graphviz[15].

## Описание и обоснование выбора состава технических и программных средств

* + 1. Описание выбора состава технических средств

Минимальные и рекомендуемые свойства компьютера:

1. Процессор Intel Core i3 с частотой 2000 МГц или более быстрый (рекомендуется не менее 3000 МГц) с подходящей материнской платой.
2. Не менее 1 ГБ оперативной памяти (рекомендуется не менее 4 ГБ).
3. Не менее 1 ГБ свободного места на жестком диске.
4. Периферия для персонального компьютера, а также клавиатура.
5. Видеокарта и монитор, способные воспроизводить графическое отображение работы программы с разрешением не менее 800х600 точек.
   * 1. Обоснование выбора состава технических средств

Выбор состава технических средств обусловлен использованием ОС Microsoft Windows XP и новее, а также комфортными условиями для быстродействия работы программы.

* + 1. Описание выбора состава программных средств

Для запуска и работы программы требуется компьютер, со следующим предустановленным программным обеспечением:

1. операционная система Microsoft Windows 7 и новее;
2. установленная JRE 1.8 или более поздняя версия;
   * 1. Обоснование выбора состава программных средств

Версия Java 8 выбрана как версия, на которой было написано приложение.

Microsoft Windows 7 выбрана как последняя из поддерживаемых ОС компанией Microsoft.

# ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ

## Предполагаемая потребность

Полученные, в ходе работы над продуктом, методы могут быть применены в индустрии для глубинного анализа процессов (см. терминологию в Приложении 1) тестовых систем.

* 1. **Ориентировочная экономическая** эффективность

В рамках данной работы расчёт экономической эффективности не предусмотрен.

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

На момент начала разработки на отечественном и зарубежном рынке не было выявлено аналогичных продуктов.

# ИСТОЧНИКИ, ИСПОЛЬЗОВАННЫЕ ПРИ РАЗРАБОТКЕ

1. ГОСТ 19.101-77 Виды программ и программных документов. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
2. ГОСТ 19.102-77 Стадии разработки. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
3. ГОСТ 19.103-77 Обозначения программ и программных документов. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
4. ГОСТ 19.104-78 Основные надписи. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
5. ГОСТ 19.105-78 Общие требования к программным документам. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
6. ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
7. ГОСТ 19.201-78 Техническое задание. Требования к содержанию и оформлению. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
8. ГОСТ 19.603-78 Общие правила внесения изменений. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
9. ГОСТ 19.604-78 Правила внесения изменений в программные документы, выполненные печатным способом. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
10. Anna Kalenkova. Learning High-Level Process Models from Event Data // Anna Kalenkova. –Eindhoven: Technische Universiteit Eindhoven, 2018. Proefschrift.
11. ProM Tools [Электронный ресурс] / Режим доступа: http://promtools.org/doku.php, свободный (дата обращения 09.05.19)
12. K-means clustering [Электронный ресурс] / Режим доступа: https://en.wikipedia.org/wiki/K-means\_clustering, свободный (дата обращения 09.05.19)
13. Damerau–Levenshtein distance [Электронный ресурс] / Режим доступа: https://en.wikipedia.org/wiki/Damerau%E2%80%93Levenshtein\_distance, свободный (дата обращения 09.05.19)
14. Задача о редакционном расстоянии, алгоритм Вагнера-Фишера [Электронный ресурс] / Режим доступа: http://bit.ly/neercIfmoWagnerFisher, свободный (дата обращения 09.05.19)
15. Graphviz [Электронный ресурс] / Режим доступа: https://ru.wikipedia.org/wiki/Graphviz, свободный (дата обращения 09.05.19)

# ПРИЛОЖЕНИЕ 1

**ТЕРМИНОЛОГИЯ** Таблица 1

|  |  |  |
| --- | --- | --- |
| **Термин (рус)** | **Термин (англ.)** | **Определение** |
| **Глубинный анализ процессов** | **Process mining** | Область исследований, занимающаяся построением моделей процессов по логам событий. |
| **Трейс** | **Trace** | Цепочка событий системы отслеживания ошибок |
| **Центроид** | **Centroid** | Центр кластера |

# ПРИЛОЖЕНИЕ 2

## Описание и функциональное назначение классов

Таблица 2. Название классов и их функциональное назначение (описание).

|  |  |
| --- | --- |
| Класс | Назначение |
| ClusterSystemBuild | Основной класс приложения, где происходит определение кластеров, построение модели для визуализации и все сопутствующие вычисления. Связывает остальные классы приложения. |
| Worker | Класс, реализующий исполнителя событий лога. |
| Event | Класс, реализующий событие лога. |
| ReadWriter | Класс, реализующий ввод и вывод данных. |
| IModelFunction | Интерфейс, реализующий функцию модели кластерного разбиения. |
| EffectiveWorker | Класс, реализующий функцию модели кластерного разбиения разработчиков в зависимости от их эффективности. |
| ProgrammerTester | Класс, реализующий функцию модели кластерного разбиения разработчиков в зависимости от рода их активности (тестировщик – программист) |
| IDistMethod | Интерфейс, реализующий метод рассчета расстояния между трейсами исполнителей. |
| DamerauLevenshteinDistMethod | Класс, реализующий метод рассчета расстояния между трейсами исполнителей, как расстояния Дамерау-Левенштейна. |
| ModelBasedDistMethod | Класс, реализующий метод рассчета расстояния между трейсами исполнителей, как абсолютную разницу между значениями функций заданной модели. |
| ICentroidsFirstPick | Интерфейс, реализующий методы выбора первоначальных центроидов. |
| RandomCentroidsFirstPick | Класс, реализующий случайный метод выбора первоначальных центроидов. |
| ModelBasedCentroidsFirstPick | Класс, реализующий метод выбора первоначальных центроидов, основанный на значениях функции заданной модели. |

## Описание и функциональное назначение полей и методов классов

Таблица 3. Описание и функциональное назначение полей класса ClusterSystemBuild

|  |  |  |  |
| --- | --- | --- | --- |
| Имя | Тип | Модификаторы | Назначение |
| modelFunction | IModelFunction | private | Функция модели разбиения |
| distMethod | IDistMethod | private | Функция рассчета расстояния между трейсами |
| centroidsFirstPick | ICentroidFirstPick | private | Функция выбора первоначальных центроидов |
| workers | Map<String, Worker> | private | Коллекция соединяющая имена разработчиков и их объекты типа Worker |
| workersNames | String[] | private | Массив имен разработчиков |
| statisticsParams | String[] | private | Массив параметров – имен событий, частоту появления которых собирает каждый из объектов типа Worker |
| k | int | private | Количество кластеров, на которое происходит разбиение исполнителей |
| readWriter | ReadWriter | private | Объект для ввода/вывода информации |
| dotClusterModel | Dot | private | Graphviz модель кластеров для визуализации |

Таблица 4. Описание и функциональное назначение методов класса ClusterSystemBuild

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя | Модификаторы | Тип возвращаемого значения | Аргументы | Назначение |
| ConstructClusters | private | Dot | PluginContext context, XLog log, CSVFile config | Главный метод класса. Возвращает Graphviz модель для визуализации кластеров. |
| buildClusters | private | void | package | Реализует метод кластеризации k-средних |
| pickConfigs | private | void | String[] configs | Выбирает методы конфигурации программы, считанные из входного файл конфигурации |
| calcFunctionValueForEveryWorker | private | void | package | Вычисляет значение функции для каждого из разработчиков |
| calcDist | private | int | Worker workerA, Worker workerB | Вычисляет расстояние между двумя трейсами разработчиков. |
| calcDistBetweenWorkers | private | Integer[][] | package | Вычисляет расстояние от каждого разработчика к каждому. |
| calcDistFromAllWorkersToTheirCentroids | private | Integer[][] | Worker[] centroids | Вычисляет расстояния от каждого из разработчиков до их центроидов. |
| determineClosestCentroidForEachWorker | private | void | Integer[][] dist | Для каждого разработчика определяет ближайший к нему центроид. |
| centroidsDifferent | private | boolean | Worker[] centroids, Worker[] newCentroids | Определяет различаются ли два переданных массива разработчиков. |
| determineNewCentroids | private | Worker[] | package | Возвращает разработчиков, которые будут являться новыми центроидами. |

Таблица 5. Описание и функциональное назначение полей класса Worker

|  |  |  |  |
| --- | --- | --- | --- |
| Имя | Тип | Модификаторы | Назначение |
| name | String | package | Имя разработчика |
| trace | Vector<Event> | private | Трейс действий разработчика |
| clusterNum | int | package | Номер кластера, к которому принадлежит разработчик |
| statistics | Map<String, Integer> | private | Коллекция, хранящая значения собранных статистик |
| functionValue | Integer | package | Поле, хранящая значение функции модели данного разработчика |

Таблица 6. Описание и функциональное назначение методов класса Worker

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя | Модификаторы | Тип возвращаемого значения | Аргументы | Назначение |
| Worker | public |  | Event event, String[] statisticsParams | Конструктор для инициализации разработчика, принимает событие и массив имен событий, подсчитывать статистику по которым будет разработчик. |
| getStatistics | package | Integer | String key | Возвращает статистику разработчика по имени события (количество раз, которое оно возникало) |
| addevent | package | void | Event event | Добавляет события в трейс разработчика |
| getTrace | package | Vector<Event> |  | Возвращает трейс действий разработчика |

Таблица 7. Описание и функциональное назначение полей класса Event

|  |  |  |  |
| --- | --- | --- | --- |
| Имя | Тип | Модификаторы | Назначение |
| workerName | String | package | Имя исполнителя события |
| activity | String | package | Название события |
| clusterNum | int | package | Номер кластера, к которому принадлежит программиста |
| statistics | Map<String, Integer> | private | Коллекция, хранящая значения собранных статистик |
| functionValue | Integer | package | Поле, хранящая значение функции модели данного разработчика |

Таблица 8. Описание и функциональное назначение методов класса Event

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя | Модификаторы | Тип возвращаемого значения | Аргументы | Назначение |
| Worker | public |  | Event event, String[] statisticsParams | Конструктор для инициализации разработчика, принимает событие и массив имен событий, подсчитывать статистику по которым будет разработчик. |
| getStatistics | package | Integer | String key | Возвращает статистику программиста по имени события (количество раз, которое оно возникало) |
| addevent | package | void | Event event | Добавляет события в трейс программиста |
| getTrace | package | Vector<Event> | package | Возвращает трейс действий программиста |

Таблица 9. Описание и функциональное назначение полей класса ReadWriter

|  |  |  |  |
| --- | --- | --- | --- |
| Имя | Тип | Модификаторы | Назначение |
| colors | String[] | private | Массив имен цветов для окраски вершин графа |

Таблица 10. Описание и функциональное назначение методов класса ReadWriter

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя | Модификаторы | Тип возвращаемого значения | Аргументы | Назначение |
| writeWorkersStatisticsToCSV | package | void | String filePath, Map<String, Worker> workers, String[] workersNames | Выводит значения статистик, которые собрал программист в .csv файл |
| writeClustersGraphvizFile | package | void | String filePath, Map<String, Worker> workers, String[] workersNames, Worker[] centroids, Integer[][] distToCentroids | Выводит Graphviz граф, содержащий кластеры исполнителей, в формате .dot в файл по пути filePath |
| writeWorkersGraphvizFile | package | void | String filePath, Map<String, Worker> workers, String[] workersNames | Выводит Graphviz граф, содержащий трейсы всех исполнителей, в формате .dot в файл по пути filePath |
| getNodeName | package | String | Worker worker | Возвращает имя программиста в кавычках для последующей записи в .dot файл |
| parseWorkers | package | Map<String, Worker> | XLog log, String[] statisticsParams | Возвращает коллекцию программистов, считанную из Log файла |
| getConfigs | package | String[] | CSVFile file | Возвращает массив строк, содержащий конфигурацию программы, считанной из .csv файла |

## 

Таблица 11. Описание и функциональное назначение методов класса EffectiveWorker

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя | Модификаторы | Тип возвращаемого значения | Аргументы | Назначение |
| function | public | int | Worker worker | Вычисляет значение функции эффективности программиста |

Таблица 12. Описание и функциональное назначение методов класса ProgrammerTester

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя | Модификаторы | Тип возвращаемого значения | Аргументы | Назначение |
| function | public | int | Worker worker | Вычисляет значение функции, значение которой определяет тестировщиком или разработчиком является программист |

Таблица 13. Описание и функциональное назначение методов класса DamerauLevensteinDistMethod

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя | Модификаторы | Тип возвращаемого значения | Аргументы | Назначение |
| dist | public | int | Worker workerA, Worker workerB | Вычисляет расстояние между трейсами, как расстояние Дамерау-Левенштейна, методом Вагнера-Фишера |

Таблица 14. Описание и функциональное назначение методов класса ModelBasedDistMethod

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя | Модификаторы | Тип возвращаемого значения | Аргументы | Назначение |
| dist | public | int | Worker workerA, Worker workerB | Вычисляет расстояние между трейсами, как абсолютную разницу между значениями функций модели |

Таблица 15. Описание и функциональное назначение методов класса RandomCentroidFirstPick

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя | Модификаторы | Тип возвращаемого значения | Аргументы | Назначение |
| pickCentroids | public | Worker[] | int k, Map<String, Worker> workers, String[] workersNames | Выбирает первоначальные центроиды случайным образом |

Таблица 16. Описание и функциональное назначение методов класса ModelBasedCentroidFirstPick

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя | Модификаторы | Тип возвращаемого значения | Аргументы | Назначение |
| pickCentroids | public | Worker[] | int k, Map<String, Worker> workers, String[] workersNames | Выбирает первоначальные центроиды, как квантили фунции заданной модели |

# ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Изм. | Номера листов (страниц) | | | | Всего листов (страниц) в документе | № документа | Входящий № сопроводительного документа и дата | Подпись | Дата |
| измененных | замененных | новых | аннулированных |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |