**Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«Саратовский государственный технический университет имени  
Гагарина Ю.А.»**

*На правах рукописи*

Петров Андрей Сергеевич

**МАТЕМАТИЧЕСКИЕ МОДЕЛИ И МЕТОДЫ ИЗВЛЕЧЕНИЯ ТЕРМИНОВ ИЗ РУССКОЯЗЫЧНЫХ ТЕКСТОВ**

*(наименование темы научно-квалификационной работы)*

|  |
| --- |
| 09.06.01 Информатика и вычислительная техника |
| *(код и наименование направления) по направленности* |
| Математическое моделирование, |
| численные методы и комплексы программ |
| *(наименование направленности)* |

Научно-квалификационная работа (диссертация)

Научный руководитель:

д.ф.-м.н., профессор,

*(ученая степень, ученое звание)*

проф. каф. ИКСП

*(должность)*

Шульга Татьяна Эриковна

*(ФИО полностью)*

Саратов 2019

[ВВЕДЕНИЕ 3](#__RefHeading___Toc24382_1389336206)

[1. Анализ моделей и методов, применяемых при оценке удобства использования программного обеспечения 12](#__RefHeading___Toc24384_1389336206)

[2. Математические модели и методы, предназначенные для анализа данных активности пользователей программного обеспечения 13](#__RefHeading___Toc24398_1389336206)

[2.1. Понятие и виды активности пользователей программной системы 13](#__RefHeading___Toc24400_1389336206)

[2.2. Модель онтологии предметной области удобства использования 16](#__RefHeading___Toc24402_1389336206)

[2.2. Модель активности пользователей 22](#__RefHeading___Toc24404_1389336206)

[2.2.\*. Данные активности пользователей 23](#__RefHeading___Toc24408_1389336206)

[3. Разработка комплекса проблемно-ориентированных программ для сбора, визуализации и анализа данных активности пользователей 26](#__RefHeading___Toc24420_1389336206)

[3.1. Анализ требований и проектирование 26](#__RefHeading___Toc24422_1389336206)

[3.2. Разработка 31](#__RefHeading___Toc24424_1389336206)

[3.3. Тестирование 37](#__RefHeading___Toc24426_1389336206)

[4. Реализация, проверка и апробация разработанных математических моделей и методов 41](#__RefHeading___Toc24428_1389336206)

[4.1. Наполнение онтологии экземплярами 41](#__RefHeading___Toc24430_1389336206)

[4.2. Построение тепловой карты 43](#__RefHeading___Toc24432_1389336206)

[ЗАКЛЮЧЕНИЕ 46](#__RefHeading___Toc24434_1389336206)

[СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ 49](#__RefHeading___Toc24436_1389336206)

# ВВЕДЕНИЕ

**Актуальность темы.** Термин — слово или словосочетание на естественном языке, описывающее понятие определенной предметной области. Извлечение терминов из текста — это процесс выделения из текста и распознавания ключевых слов и терминов, описывающих понятия определенной предметной области. Извлечение терминов из текста является важным этапом в ряде задач, связанных с обработкой текстов предметной области, среди которых — информационный поиск, машинный перевод, классификация и кластеризация документов, построение онтологий, глоссариев и тезаурусов.

Существует большое количество методов автоматического извлечения терминов из текста на естественном языке. Часть из них создавалась с учетом особенностей ряда таких языков, относящихся к индоевропейским языкам романо-германской группы, как английский и французский, и не учитывает особенности грамматики и морфологии русского языка. Кроме того, эффективность работы данных методов остается низкой и варьируется в зависимости от предметной области и стиля исходного текста. Также большое количество методов извлечения терминов из текстов для обработки используют входные данные, размеченные вручную. Данные особенности и недостатки ограничивают их область применимости.

Большая часть существующих методов относятся к категории статистических методов извлечения терминов из текста. Исследованиям в области применения статистических критериев для выбора ключевых слов и понятий посвящены работы Roberto Navigli, Paola Velardi, Katerina Frantzi, Sophia Ananiadou, Hideki Mima, David A. Evans, Robert G. Lefferts. Основным критерием, использующихся для процесса извлечения терминов из текста, в данных работах является критерий частоты вхождения кандидатов в рассматриваемую коллекцию текстовых документов. На таком принципе построена работа методов Domain Consensus [1], C-Value [2], TF-IDF [3]. Большая часть понятий в предметных областях являются многословными терминами. Однако в текстах, написанных на естественных языках, вне зависимости от предметной области такие понятия встречаются значительно реже по сравнению с однословными понятиями. Существующие методы по-разному компенсируют данный эффект. Так методы, основывающиеся на подсчете меры ассоциации, учитывают вероятность совместного употребления слов в составе определенного термина. К таким метрикам относятся, в частности, взаимная информация (Mutual Information, MI), логарифмическое правдоподобие (Loglikelihood Ratio). В ряде методов учитывается вложенность понятий предметной области. К таким, например, относится C-Value [2]. Вложенные термины (nested terms) - это понятия, содержащиеся в исходном тексте как по отдельности, так и в составе других понятий. Другим подходом к улучшению результатов работы методов извлечения ключевых слов из текста является учёт контекста вхождений. Одним из методов, использующим данный подход, является, например, NC-Value [2].

Еще одним способом улучшения результатов извлечения многословных терминов является частота вхождений слов и словосочетаний во внешнюю по отношению к рассматриваемому тексту коллекцию документов с точки зрения предметной области. Причем документы подбираются таким образом, чтобы их тематика не ограничивалась определенной предметной областью. То есть в случае обработки исходного текста военно-исторической предметной области подбирается корпус новостей или художественной литературы. Текстовый корпус – это сформированная по определенным правилам, структурированная коллекция текстов. К методам, учитывающими данную метрику, относятся, в частности, Domain Relevance [5], Weirdness [7], Relevance [7], Domain Pertinence [8].

Часть методов извлечения терминов из текста применяют внешние ресурсы: например, поисковые машины [9, 10] или энциклопедию Википедия [11, 12]. Большая часть подобных методов не использует заранее подготовленную коллекцию текстовых документов, применяя исключительно внешние ресурсы для поиска и извлечения терминов. Однако, при таком подходе без предварительного составления текстового корпуса достаточно сложно добиться высокой точности извлечения перечня терминов.

Таким образом, значительная часть существующих методов извлечения терминов из текста, основывающихся на использовании статистических метрик для выбора понятий, ограничивается текстами выбранной предметной области, которые обычно не являются размеченными текстовыми корпусами и в связи с этим не содержат в себе необходимого объема информации для автоматического извлечения терминов. Ряд методов для решения этой проблемы используют внешние ресурсы, такие как поисковые машины или контрастные корпуса текстов других предметных областей. Текстовые документы, в большинстве своем, не имеют структуры и позволяют использовать только статистическую информацию о частоте встречаемости слов и словосочетаний в тексте без учета особенностей рассматриваемой предметной области.

В качестве модели представления исходных текстовых данных большая часть статистических методов извлечения терминов из текста использует модель «мешок слов» (англ. Bag of words). Данная модель представляет текстовый документ в виде множества слов, его составляющих, а также их частот встречаемости. Разновидностью модели Bag of words является частотная модель текста, в которой каждому слову соответствует весовой коэффициент, который определяется в зависимости от выбранной метрики: например, частоты вхождения слова в документе TF (англ. Term frequency), логарифма частоты вхождения слова Log TF или обратной частоты документов IDF (англ. - Inverse document frequency). Однако, вышеописанные методы не учитывают порядок слов и могут быть использованы для извлечения лишь однословных понятий из текста.

Для решения этой проблемы вводится модель коллокации N-грамма (англ. N-gram). Коллокация — словосочетание, являющееся синтаксически и семантически целостной единицей. Данная модель позволяет представить словосочетания из N слов, при N=2 словосочетание имеет название биграммы, при N=3 — триграммы. Таким образом, для извлечения многословных терминов возможно использование частотной модели текста вместе с моделью коллокаций N-грамма. Однако, данные модели не принимают во внимание грамматические особенности русского языка, с учётом которых возможно повышение качества результатов методов извлечения терминов из текста.

Вышесказанное определило актуальность настоящей работы, а также, следующие из неё цели и задачи.

Цель работы – cодержательная постановка задачи автоматического извлечения терминов из текста, разработка математической модели русскоязычного текстового документа, которая может быть использована для рассматриваемой задачи, проведение исследования статистических методов выделения понятий на текстах исторической предметной области

Задачи работы. Для достижения поставленной цели были поставлены и решены следующие основные задачи:

1. Исследовать существующие математические модели и методы применяемые при автоматическом извлечении терминов из текста.
2. Разработать математическую модель русскоязычного текстового документа.
3. Создать комплекс проблемно-ориентированных программ, реализующий разработанную математическую модель и методы автоматического извлечения терминов из текста.

**Объект исследования** – процесс извлечения терминов из русскоязычных текстовых документов.

**Предмет исследования** – математические модели и методы, применяемые при выделении ключевых слов и понятий.

**Методы исследования**: аппарат теории множеств, методы обработки многомерных данных, методы параллельного и объектно-ориентированного программирования, методы статического анализа.

**Научная новизна работы** (соответствует пунктам 1, 4 паспорта специальности 05.13.18) заключается в следующем:

Разработана математическая модель текстового документа на русском языке, учитывающая грамматические особенности русского языка, описанная в форме терминологии с использованием аппарата теории множеств. С помощью разработанного комплекса проблемно-ориентированных программ исследована применимость методов автоматического извлечения терминов из текста на текстах военно-исторической предметной области.

**Практическая значимость работы** обусловлена созданием комплекса проблемно-ориентированных программ для извлечения терминов из коллекции текстовых документов.

**Основные положения и результаты**, выносимые на защиту:

1. Математическая модель предметной области текстового документа на русском языке, учитывающая грамматические особенности русского языка, описанная в форме терминологии с использованием аппарата теории множеств.
2. Разработанный комплекс проблемно-ориентированных программ «Программный комплекс для извлечения терминов из текстов», на основе предложенной модели.

**Достоверность результатов**.

Достоверность и обоснованность результатов работы определяется корректной постановкой задач, применением методов статистического анализа, обработки многомерных данных. Результаты исследования подтверждены вычислительными экспериментами и практической апробацией.

**Апробация работы**. По основным результатам работы сделано 4 докладов на 4 международных, всероссийских и региональных конференциях:

1. Международная научно-практическая конференция «Проблемы управления в социально-экономических и технических системах» (ПУ-2016).
2. Международная научная конференция «Информационно-коммуникационные технологии в науке, образовании и производстве» (ICIT-2016).
3. Международная научная конференция «Информационно-коммуникационные технологии в науке, образовании и производстве» (ICIT-2017).
4. Международная научная конференция «Проблемы управления, обработки и передачи информации» (УОПИ-2017).

**Публикации**

Результаты работы опубликованы в 5 изданиях, 1 из которых являются изданиями, рекомендованными ВАК, 4 индексируются в базе РИНЦ. Имеется свидетельство о государственной регистрации программы для ЭВМ №2017663283 от 28 ноября 2017 г.

**Соответствие темы диссертации требованиям паспорта специальностей научных работников.** Диссертационная работа выполнена в соответствии с паспортом специальности 05.13.18 – Математическое моделирование, численные методы и комплексы программ, п. 1. Разработка новых математических методов моделирования объектов и явлений; п. 4. Реализация эффективных численных методов и алгоритмов в виде комплексов проблемно-ориентированных программ для проведения вычислительного эксперимента.

**Структура и объем работы.** Научно-квалификационная работа состоит из введения, трех глав, заключения, списка использованной литературы и приложения. Работа содержит \_\_ страниц, включая \_\_ рисунка, библиографический список из \_\_ наименований.

# 1. Анализ моделей и методов, применяемых при оценке удобства использования программного обеспечения

В первой главе приведен обзор истории развития научно-прикладной дисциплины удобства использования, а также, моделей, методов и инструментов, применяемых при оценке удобства использования программного обеспечения, включая модели онтологий, связанных с исследованиями в области пользовательских интерфейсов и оценки удобства использования.

Обзор показывает, что существует довольно обширный пласт исследований, в области анализа шаблонов поведения пользователей. Данная тематика имеет большой потенциал, в виду очевидной экономии ресурсов и времени при автоматическом анализе данных активности пользователей с целью быстрого обнаружения мест требующих детального исследования.

# 2. Математические модели и методы, предназначенные для анализа данных активности пользователей программного обеспечения

## 2.1. Понятие и виды активности пользователей программной системы

Проведенный анализ показал, что различные методы, эвристики и подходы, оценки юзабилити сформированы основываются на понимании того, как пользователь ведет себя при взаимодействии с интерфейсом, какие ошибки совершает. Как показывает практика, на этапе проектирования можно предусмотреть очень многие нюансы, но не все. Для детального понимания, насколько пользователь может продуктивно, эффективно и с должной степенью комфорта, работать с компьютерной системой, необходимо анализировать его реальное поведение при работе с компьютерной системой посредством пользовательского интерфейса. Поэтому, например, юзабилити-тестирование в последнее десятилетие стало обязательным этапом в процессе разработки ПО, особенно при разработке крупных компьютерных систем [33].

В общемировой практике, к сожалению, не сложилось единой классификации для всех существующих видов активности пользователя. В первую очередь это связано с тем, что можно выделить огромное количество видов активности пользователя специфичных лишь для конкретного приложения, его типа (настольное, мобильное и т.д.) или сферы применения. Так, неотъемлемая для интернет-магазинов активность «покупка товара», будет отсутствовать на сайте для ведения блогов. В социальной сети пользователь может добавить другого пользователя в свой список друзей, тем самым выполнив активность «добавление друга», но для ПО управления томографом такая активность будет неприменима. В профессиональном сообществе устоялось мнение, что при необходимости классификации видов активности, проще и рациональнее ограничить их перечень конкретным типом приложения, для которого выполняется сбор данных. Применительно к веб-приложениям, к таковым наиболее часто используемым можно отнести:

* вход в систему (авторизация, аутентификация), выход из нее;
* клик мышью, нажатие сенсорного экрана (для мобильных версий);
* перемещение курсора мыши;
* выделение, копирование и вставка текста;
* заполнение и отправка форм;
* навигация по страницам;
* прокрутка страниц;
* нажатие клавиш.

Данные различных видов активности пользователя, позволяют провести анализ различных аспектов приложения. Карта кликов, например, позволяет получить следующие выводы:

* эффективность использования приложения пользователем;
* общая характеристика использования приложения;
* популярность разделов.

Почти все существующие онлайн-инструменты для сбора статистики в веб-приложениях позволяют собирать подобную информацию. Чаще всего, этого достаточно, чтобы понять, как пользователь взаимодействует с приложением и найти основные недочеты в существующем интерфейсе. Специфичные виды активности пользователя (например, «покупка товара», «добавление друга» и т.п.) обычно отслеживаются для особых случаев и с помощью собственных доработок в компьютерной системе.

При анализе взаимодействия пользователя с приложением, может быть полезна любая информация о любой активности пользователя. Чем большим объёмом информации владеет эксперт, тем больше факторов он сможет учесть. Однако, очень важно, что информация так же должна быть пригодна для последующей обработки и использования. Например, при юзабилити-тестировании иногда ведут видеозапись происходящего на экране и/или поведения пользователя. Это позволяет отследить практически все виды активности, но последующая обработка таких видеозаписей крайне трудоемка, не говоря уже о сложностях с подготовкой испытуемых. Методика видеозаписи обеспечивает полноту собираемой информации, но не обеспечивает способа ее хранения в виде статистических данных, с которыми можно было бы производить дальнейшие операции.

Важна так же возможность сбора данных активности пользователя без использования специальных лабораторий. Кэтрин Томпсон, вместе со своими коллегами, в 2004 году опубликовала исследование «Here, there, anywhere: Remote usability testing that works» [34], в котором подробно рассказывается о разнице между лабораторными исследований и сбором статистики в реальных условиях. По результатам исследований был сделан вывод, что даже при самой тщательной подготовке, испытуемые будут вести себя в лабораторных условиях иначе, нежели в реальной жизни. То есть, некоторые ошибки просто могут быть не обнаружены при лабораторных исследованиях.

Особо можно выделить информацию о перемещении взгляда пользователя при работе с приложением. Во-первых, сбор информации по данному виду активности можно применять фактически для всех типов и видов приложений. Во-вторых, взгляд пользователя на ту или иную точку экрана не является непосредственным его действием, но может являться стимулом для совершения следующей активности. Для сбора информации о данном виде активности используется методика окулографии (айтрекинг). Она требует дорогостоящего оборудования и трудновыполнима вне лабораторных условий, но зато позволяет собрать всю необходимую информацию в формализованном виде. Кроме оценки юзабилити, окулография применяется при анализе раскладки продуктов в магазине, дизайна рекламных объявлений и т.д.

На основе вышеописанного, было проведено обобщение наиболее формализованных и универсальных видов активности пользователей настольных приложений, в следующие группы:

* нажатие указательного устройства ввода или его кнопок, например, мыши (щелчок, клик, англ. click), тачпада или сенсорного экрана);
* изменение координат указательного устройства ввода (перемещение курсора мыши, движение пальца по сенсорному экрану);
* прокрутка (скролл, англ. scroll) интерфейса или его областей, например, содержимого окна целиком или списка элементов;
* навигация по разделам приложения (окнам, вкладкам);
* ввод данных, манипуляции с полями для ввода данных (выпадающими списками, текстовыми, логическими и т.п.);
* перемещение взгляда пользователя;
* прочие виды активности, специфичные для типа/вида приложения.

Данный перечень можно считать применимым к любым видам настольных приложений, при необходимости детализируя перечень прочих видов активности.

## 2.2. Модель онтологии предметной области удобства использования

Разработана модель предметной области «Удобство Использования ПО», представленная в форме онтологии:

где:

– термины предметной области,

– множество классов, описывающих условия использования (классы «Session», «Device», «User» и т.д.);

– множество классов, описывающих программный интерфейс (классы «Region», «Variation», «Image» и т.д.);

– множество классов, описывающих активность пользователей (классы «Event», «MouseActionEvent» и т.д.);

– отношения между терминами заданной предметной области, где возможны следующие варианты отношений:

– множество объектных свойств;

, где и

(свойства «wasPerformedOn», «wasUsedIn», «wasPerformedBy», «performed» и т.д.);

(свойства «contains», «isContainedIn» и т.д.);

, при этом потомок

– множество свойств данных (свойства «hasStartDateTime», «hasWidth», «hasDpiX» и т.д.);

, где и

,

– значение встроенного типа (integer, float, binary, string).

– множество функций интерпретации (аксиоматизации), заданных на терминах и/или отношениях онтологии.

Онтология предметной области «Удобство использования программного обеспечения» в первую очередь предназначена для фиксации данных о взаимодействии пользователя с программной системой посредством графического пользовательского интерфейса.

Описаны основные понятия, используемые в рассматриваемой предметной области и взаимосвязи между ними, т.е. классы и свойства предлагаемой онтологии. Заметим, что на рисунке в виде дуг отображены только основные объектные свойства онтологии (описывающие отношения между экземплярами классов, англ. Object Properties). Свойства данных (описывающие отношения экземпляров класса с литеральными значениями, англ. класса DataType Properties) описаны ниже.

Существующие онтологии позволяют описать интерфейс, например, [9,10] или способы взаимодействия пользователя с интерфейсом [11]. Однако, они не пригодны для последующего накопления данных о самом взаимодействии. Обоснуем данный вывод, приведя краткий обзор этих онтологий.

Исследователи Энн Блэндфорд (Ann Blandford) и Томас Грин (Thomas Green) еще в 1997 году предложили онтологическую модель построения эскизов (англ. Ontological Sketch Models, OSM) [11]. Они ставили перед собой цель разработать такой подход к оценке удобства использования ПО, который был бы основан на теоретических результатах исследований научных сообществ, но мог быть использован командами дизайнеров в промышленности. При использовании описанного ими подхода эксперт формирует структурированное, но простое представление анализируемого программного обеспечения на основе упрощенной модели онтологии. Онтология OSM покрывает три аспекта дизайна системы: сущности (англ. entities), действия (англ. actions), взаимосвязи (англ. relationships). Ниже они рассмотрены на примере текстового редактора.

Сущность – понятие или объект, который пользователь должен знать (символ, слово, параграф, ширина колонки) и который обладает следующими свойствами:

• атрибуты (англ. аttributes) – дополнительные характеристики которыми обладает сущность, например, символ имеет шрифт, размер;

• доступность (англ. accessibility) – на уровне понимания пользователя, уровне устройства или системы и общие;

• релевантность (англ. relevance) – релевантность сущности по отношению к определенной области и/или к устройству, например, слово релевантно к области написания текста, а полоса прокрутки (англ. scrollbar) релевантна к текстовому редактору;

• видимость (англ. persistent visibility) – может ли пользователь увидеть сущность, существует она условно или прозрачна;

• маскировка (англ. disguise) – обладает ли сущность понятным названием или символикой.

Действие – то, что пользователь может совершать. Действие имеет следующие свойства:

• название действия;

• сущность(и), над которой(ми) совершается действие;

• эффект (англ. effect), получаемый после совершения действия;

• контекст (англ. context), контекстная информация.

Взаимосвязь – связь между сущностями, обладает свойствами:

• тип (англ. type) – «состоит из», «влияет», «ограничивает» или любой другой;

• сущность(и) – две или более сущности, имеющие взаимосвязь.

Валерия Грибова в своей работе «Модель онтологии предметной области “Графический пользовательский интерфейс”, опубликованной в 2005 году [10], предложила модель значительно более сложную, чем OSM, т.к. основная идея её использования формирование декларативной модели пользовательского интерфейса на основе универсальных моделей онтологий и последующая автоматическая генерация исполнимого кода интерфейса. Для описания модели пользовательского интерфейса разработана подробная модель онтологии «графический пользовательский интерфейс», которая описывает графические интерфейсные элементы, их свойства и связь друг с другом для формирования диалога с пользователем, основанном на экранных формах. Базовая универсальная онтология включает в себя более 50 классов, где классы содержат 10 и более свойств. Среди различных видов визуальных средств графического пользовательского интерфейса (ГПИ) выделяются две основные группы – окна и оконные элементы управления, и три дополнительные – панели управления, оконные меню и вспомогательные средства. Фрагмент иерархии элементов ГПИ (классов) представлен на рис. 1.

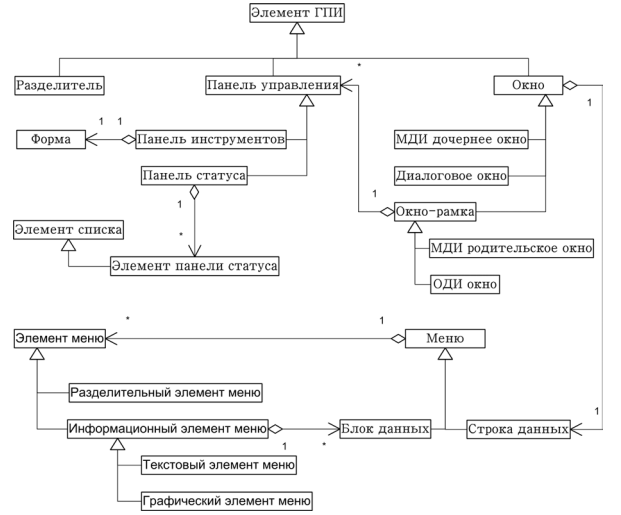


Рис. 1. Фрагмент иерархии элементов ГПИ

Элемент ГПИ – класс, описывающий общие для всех элементов ГПИ свойства, свойства: X (экранная координата левого верхнего угла элемента по оси X в пикселях, тип: целое число); Y (экранная координата левого верхнего угла элемента по оси Y в пикселях, тип: целое число); ширина (ширина элемента в пикселях, тип: целое число); высота (высота элемента в пикселях, тип: целое число); отображаемость (признак, отображается ли элемент на экране, тип: булевский); доступность (признак, доступен ли элемент для взаимодействия с пользователем, тип: булевский); меню (каждому элементу соответствует [0, 1] контекстных меню в виде блока данных, тип: блок данных) и т.д.

Окно – класс, описывающий свойства, области экрана, с помощью которой пользователь имеет возможность получить визуальное представление определенного аспекта решаемой задачи, суперкласс: «Элемент ГПИ», свойства: иконка (каждому рамочному элементу соответствует 0 или 1 небольших изображений, находящихся в левом верхнем углу экрана, которые используются для графической идентификации окна, тип: образ) и т.д.

В каталоге открытых связанных словарей (Linked Open Vocabularies) содержится онтология «Пользовательский интерфейс» (англ. user interface, UI) [9] под авторством Тимоти Джона Бернерс-Ли (англ. Timothy John Berners-Lee), английского ученого, одного из создателей всемирной паутины, автора концепции семантического веба. Онтология достаточно простая (24 класса, 27 свойств и 4 экземпляра), но содержит важные базовые понятия из предметной области пользовательского интерфейса, такие как: форма (англ. Form); поле (англ. Field) и т.д., фрагмент модели представлен на рис. 2.

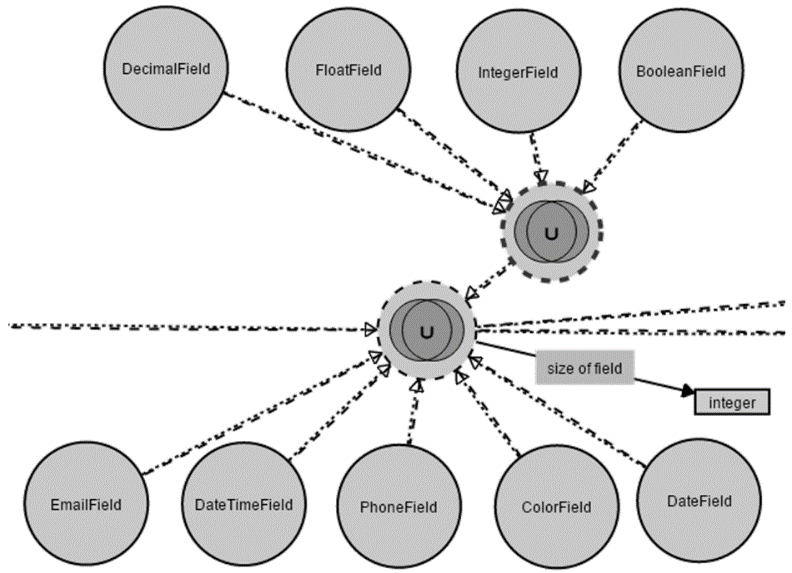


Рис. 2. Фрагмент онтологии пользовательского интерфейса

Таким образом, рассмотренные онтологии позволяют описать интерфейс анализируемого программного обеспечения, но не содержат классов и свойств, описывающих данные активности пользователей с привязкой к пользовательскому интерфейсу: движение курсора мыши, клики клавиш мыши, нажатие клавиш клавиатуры, различные формы взаимодействия с сенсорным экраном и т.п. Существуют и другие онтологии, упоминаемые в научных работах и связанные в той или иной мере с предметными областями «Пользовательский интерфейс» и «Удобство использования ПО», но не все из них удаётся найти в открытых каталогах онтологий. Онтологии, связанные непосредственно с терминами «Удобство использования» и «Данные активности пользователя», найти в открытых научных источниках не удалось. Однако рассмотренные онтологии могут служить отправной точкой при разработке общей онтологии предметной области «Удобство использования программного обеспечения».

## 2.2. Модель активности пользователей

Уровень удобства использования программного интерфейса влияет на качество всего программного обеспечения в целом [1]. Признаком недостаточного уровня удобства использования является наличие проблем взаимодействия пользователя с пользовательским интерфейсом. Они могут быть связаны либо со сложностью формулирования плана действия (принятия решений, что делать дальше, англ. articulation translation), либо с непониманием ответа системы (как изменения в интерфейсе связаны с выполненными действиями, англ. observation translation) [2].

Проблемы взаимодействия в большинство случаев возможно определить по наличию в данных активности пользователей определенных последовательностей действий (шаблонов). Для их обнаружения применяются различные методы анализа собираемых данных – как требующие ручного анализа (например, тепловые карты [3,4], англ. heat map), так и использующие алгоритмы автоматического анализа [2] на основе шаблонов, выявленных исследователями ранее [5-7]. Автоматический анализ экономит время и деньги, так как эксперты, вместо анализа всех данных, фокусируют своё внимание на отдельных областях пользовательского интерфейса, где были выявлены соответствующие шаблоны.

На данный момент в открытых научных источниках не удалось найти формализованное представление данных активности пользователей программного обеспечения. В работе представляется разработанная авторами математическая модель активности пользователей программного обеспечения. Данная модель может найти применение при оценке удобства использования пользовательских интерфейсов. Целью является максимальная формализация оценки удобства использования и формирование критериев для повышения эффективности взаимодействия пользователей с пользовательским интерфейсом.

### 2.2.\*. Данные активности пользователей

Введем несколько основных понятий, необходимых для формализации представления активности пользователей применительно к области оценки удобства использования.

Событие – факт, зафиксированный в определенный момент времени при взаимодействии определенного пользователя на определенном устройстве с программным пользовательским интерфейсом [11]. Событие обладает непустым уникальным набором атрибутов: пользователь, устройство, время, тип события (например, событие действия, командное событие) и специальные атрибуты, зависящие от типа события. Например, событие клика мыши может содержать: наименование клавиши (левая, права), тип клика (одинарный, двойной), координаты положения курсора мыши и т.д. Событие исполнения команды (командное событие) может содержать: наименование кнопки, наименование команды, способ исполнения (клик мыши, горячая клавиша, клавиша «Enter»).

Обозначим – множество всех зафиксированных событий:

,

где – отдельные события, – мощность множества .

Сессия – зафиксированный временной промежуток, в течении которого пользователь взаимодействовал с программной системой [11]. В рамках сессии накапливаются и хранятся все события в хронологическом порядке. Важно отметить, что каждое событие является уникальным и может быть включено только в одну сессию.

Определим размещение – упорядоченный набор элементов множества, либо с повторениями, либо без повторений, в соответствии с общепринятым определением [12]. Размещение будем заключать в треугольные скобки. Тогда, если обозначать элементы множества целыми числами, то размещения будут записаны в виде .

Обозначим – множество всех зафиксированных сессий:

,

где – отдельные сессии, – мощность множества .

Особо отметим, что каждое событие обладает непустым уникальным набором атрибутов, а, следовательно, является уникальным.

Сессия представляет собой размещение элементов множества *без повторений:*

,

где – отдельное событие *i*-ой сессии, – мощность размещения .

Множество формируется в результате объединения всех множеств сессий, полученных из данных активности пользователей. При этом, любое событие принадлежит хотя бы одной сессии:

*,*

где – множество всех зафиксированных событий, e – отдельное событие, S – множество всех зафиксированных сессий, s – отдельная сессия.

Аналогично расчету временных затрат сессии, оценка времени, затрачиваемого на один шаблон, сводится к сложению продолжительности его составляющих и произведению полученной суммы на количество вхождений.

# 3. Разработка комплекса проблемно-ориентированных программ для сбора, визуализации и анализа данных активности пользователей

В третьей главе рассматривается процесс разработки комплекса проблемно-ориентированных программ.

## 3.1. Анализ требований и проектирование

При проектировании подобного ПО для настольных приложений, необходимо учитывать некоторые важные отличия от веб-приложений.

Во-первых, большая часть веб-приложений имеет четкое разбиение на отдельные веб-страницы. При этом каждая веб-страница является самостоятельной структурной единицей с особым функционалом и, как следствие, особым сценарием взаимодействия с ней пользователя. Настольные приложения фундаментально имеют только один способ разделения на области (регионы) активности пользователя – это создание отдельных окон с различной функциональностью. Однако, данный способ нередко усложняет процесс работы пользователя с приложением и на практике часто реализуются однооконные приложения, со сменой области активности и открывающимися дополнительными панелями. По такому принципу реализован интерфейс ПО Microsoft Word версии 2007 и выше, где открываемые панели и вкладки содержат большое количество функционала.

Во-вторых, несмотря на то, что пользователь взаимодействует с веб-приложениями посредством веб-браузера, все модули и веб-страницы располагаются на сервере(ах). Таким образом, при анализе активности пользователя веб-приложения, практически всегда есть возможность воспроизвести внешний вид интерфейса, зная, например, URL-адрес текущей страницы, разновидность и версию веб-браузера, размеры его окна и локализацию. В отличии от веб-приложения при работе настольного приложения, модули могут располагаться где угодно (локальный диск, сетевое хранилище, удаленный доступ) и факторов, влияющих на него несравненно больше (операционная система, версии системных библиотек и библиотек платформы, пользовательские настройки и т.д.). Поэтому, воспроизвести внешний вид пользовательского интерфейса настольного приложения достаточно проблематично и рациональнее создавать его изображения (снимки) в процессе работы, например, определенных уникальных участков.

В процессе анализа предметной области, был выделен ряд следующих базовых терминов для описания программного обеспечения и его структуры.

Сессия (англ. Session) – сессия, интервал времени, когда пользователь работает (взаимодействует) с приложением. В рамках сессии накапливается и хранится вся информация об активности пользователя. Аналогичный термин применяется при описании подобного программного обеспечения для веб-приложений.

Регион (англ. Region) – область пользовательского интерфейса в пределах которой ведется отслеживание действий пользователя, например, окно целиком, либо его отдельная части. Именно изображение региона необходимо при оценке юзабилити и анализе собранных данных активности пользователя.

Активность (англ. Activity) – выполнение пользователем какого-либо отслеживаемого действия в рамках региона.

В ходе анализа аналогичных программных продуктов и проведенных исследований в области оценки юзабилити были сформулированы следующие функциональные требования для разрабатываемого программного обеспечения:

Сформулированы следующие базовые функциональные требования для разрабатываемого программного обеспечения:

* сбор данных активности пользователей в максимально возможном объеме относительно разработанной модели онтологии:
  + отслеживание нажатия левой кнопки мыши (клик);
  + отслеживание движения курсора мыши;
  + создание графических изображений регионов интерфейса;
* сохранение отслеживаемой информации в виде файла(ов);
* загрузка файлов с данными активности пользователя;
* визуализация данных активности пользователя:
  + таблица (список активности, действий);
  + тепловая карта (теплокарта, heatmap), состоящая из слоя с самой тепловой картой поверх слоя с изображением региона интерфейса, с возможностью настройки параметров (прозрачность слоя с тепловой картой; цветовая схема (палитра); дистанция градиента; дистанция перекрытия.)

Требуемый функционал рациональнее разделить на модули, что позволит следовать принципу разделения обязанностей и упростит последующее тестирование. Таким образом, была спроектирована структура ПО, состоящая из двух уровней и схемы для хранения данных активности пользователя в виде XML-файлов и файлов изображений регионов, представленная в виде диаграммы на рисунке. Опишем каждый из элементов данной диаграммы.

User Activity Client Library (UACL) – клиентский модуль для сбора данных активности пользователя, подключается к приложению, в котором необходимо выполнять сбор данных. В виду специфики разработки настольных приложений на различных платформах (WPF – .NET, SWT – Java, VCL – Delphi, и т.д.), разработать универсальную клиентскую библиотеку не представляется возможным. Поэтому, под каждую платформу необходимо будет создавать отдельную версию UACL, для учета всей ее специфики. Подобные клиентские библиотеки должны соответствовать функциональным требованиям, связанным со сбором данных активности пользователей



User Activity Data (UAD) – это файлы, генерируемые UACL-модулем в процессе работы приложения и сбора данных активности пользователя. Состоят из XML-файлов (соответствующих специальной XSD-схеме, см. приложение 1) и графических файлов (BMP, JPG, PNG), содержащих изображения регионов АП, созданные клиентскими библиотеками в процессе работы. Для предоставления определенной гибкости, при реализации UACL-модулей для различных платформ и передаче данных, XSD-схема предусматривает хранение изображений как непосредственно внутри XML-файлов, в формате base64-строки массива байт в CDATA-секции, так и виде путей к файлам изображений так же внутри CDATA-секций (рисунок 3.1.2).



Рисунок 3.1.2 – Пример XML UAD-файла

User Activity Viewer (UAV) – модуль второго уровня, по сути являющийся приложением для загрузки UAD-файлов и их визуализации в виде таблиц или тепловых карт.

Стандартный сценарий сбора данных активности пользователя в рамках спроектированной структуры ПО представляется следующим:

1. пользователь работает (взаимодействует) с настольных приложением, к которому подключен UACL-модуль;
2. действия (активность) пользователя отслеживаются библиотеками UACL и в процессе работы хранятся в памяти;
3. по завершению работы, все собранные данные и вспомогательная информация записываются в виде UAD-файлов;
4. UAD-файлы получает пользователь UAV (либо в ручном режиме они копируются/пересылаются, либо сразу помещаются в выделенное хранилище);
5. пользователь UAV импортирует UAD-файлы и в дальнейшем анализирует данные активности пользователя для решения поставленной задачи.

## 3.2. Разработка

Было принято решение для разработки использовать бесплатную версию Microsoft Visual Studio Express 2012 для Windows Desktop. Платформой была выбрана WPF версии .NET 4.0, так как она обладает богатыми возможностями построения пользовательских интерфейсов. В качестве вспомогательных библиотек использовался набор элементов управления Extended WPF Toolkit версии 1.9.

Основная часть процесса разработки велась в соответствии с принципами разработки через поведение (англ. Behavior Driven Development, BDD), когда для требуемого функционала и классов сначала пишутся автоматические модульные тесты, а уже потом код позволяющий их успешно выполнить. Это позволяет обеспечить простоту и структурированность создаваемого кода, избегая излишнего функционала.

Модуль первого уровня назван «UserActivity.CL.WPF». Он содержит два класса для сбора данных активности пользователя – «MouseClickBehavior» и «MouseMoveBehavior», которые отвечают за сбор кликов мышью и движений курсора соответственно. Классы содержат так называемые прикрепляемые свойства (англ. Attached Property):

* «IsEnabled» – флаг (булевый тип), включающий или выключающий отслеживание активности пользователя в указанном регионе;
* «RegionName» – названия региона (строковый тип), определяющий название указанного региона.

Подключение данного UACL-модуля в WPF-приложение состоит из нескольких шагов. Сначала подключается DLL-библиотека (UACL-модуль), добавляющая поддержку прикрепляемых свойств (атрибутов). Затем в XAML-разметку добавляется пространство имен, содержащее прикрепляемые свойства, которые указываются для требуемых элементов управления (рисунок 3.2.1). Полный перечень регионов заранее конфигурировать не надо, в UAD-файлах все формируется автоматически.

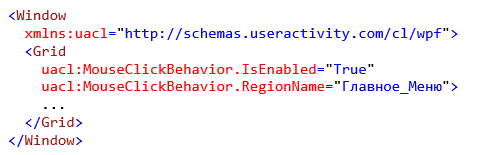


Рисунок 3.2.1 – Подключение UACL в XAML-разметке WPF-приложения

Вся активность пользователя регистрируется при помощи Singleton-сервиса «UserActivityService». Для окончательного подключения UACL-модуля и активации сбора данных активности пользователя, так же необходимо добавить вызов методов «OpenSession» и «CloseSession», для открытия и закрытия сессии соответственно, в «UserActivityService».

Классы «MouseClickBehavior» и «MouseMoveBehavior», после возникновения отслеживаемого действия, собирают всю необходимую информацию для регистрации активности и обращаются к сервису «UserActivityService». Он, в свою очередь, так же собирает служебную информацию и обращается к текущему экземпляру класса контекста данных, по-умолчанию «XmlUserActivityDataContext», который отвечает за накопление данных и последующее их сохранение в локальной файловой системе в момент закрытия сессии. За счет использования принципов контрактного программирования, «XmlUserActivityDataContext» может быть заменен другим классом контекста данных поддерживающим специальный интерфейс «IUserActivityDataContext», например, для отладки и вывода в консоль.

Модуль второго уровня, UAV, назван «UserActivity.Viewer» и представляет собой WPF-приложения, реализуемое в соответствии с шаблоном проектирования Model-View-ViewModel (MVVM). При таком шаблоне проектирования архитектура приложения разделяется на три слоя:

* модель (англ. Model, доступ к данным);
* модель отображения (англ. View Model, подготовка данных для их загрузки в элементы управления, то есть их отображение);
* отображение (англ. View, элементы управления для отображения).

Дизайн интерфейса UAV выполнен в соответствии принципами минимализма (рисунок 3.2.2). Слева располагается область для загрузки UAD-файлов и их визуализации в рамках выбранного региона, список которых автоматически наполняется при загрузке файлов.

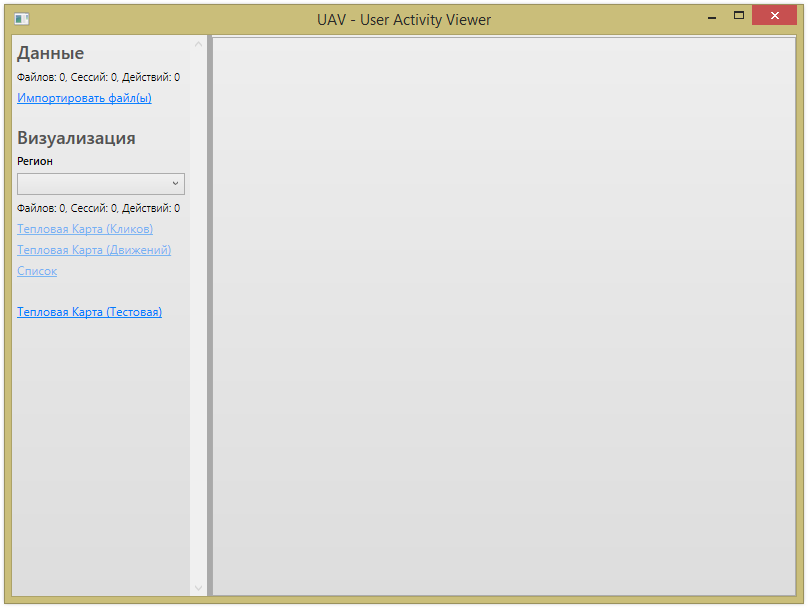


Рисунок 3.2.2 – Стартовый вид UAV

После выбора региона становятся активными ссылочные кнопки для визуализации данных. При нажатии на кнопку, справа в рабочую область добавляется новая вкладка с визуализацией для выбранного региона, которую можно закрыть, нажав символ крестика рядом с ее названием (рисунок 3.2.3).

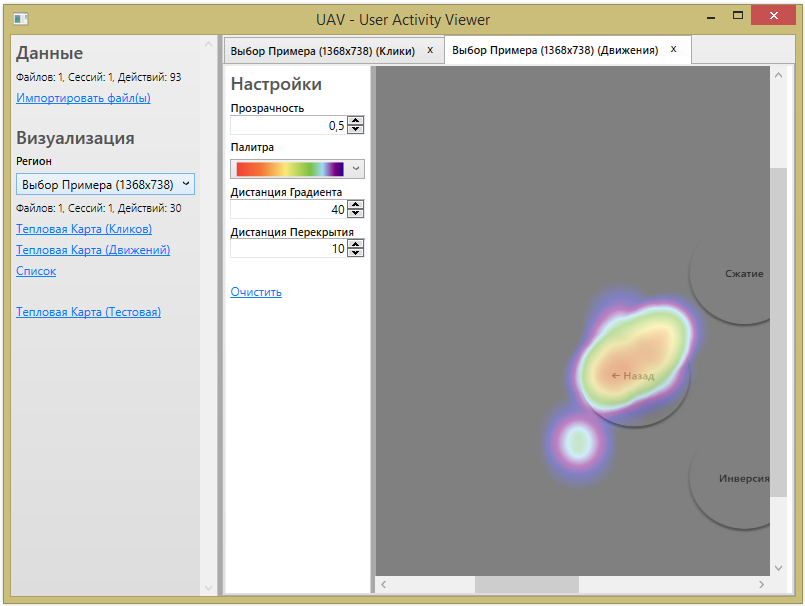


Рисунок 3.2.3 – Тепловая карта в UAV (перемещение курсора)

Вкладки для визуализации посредством тепловых карт имеют поля ввода, для возможности настройки параметров построения тепловой карты (рисунок 3.2.4), указанных в функциональных требованиях.

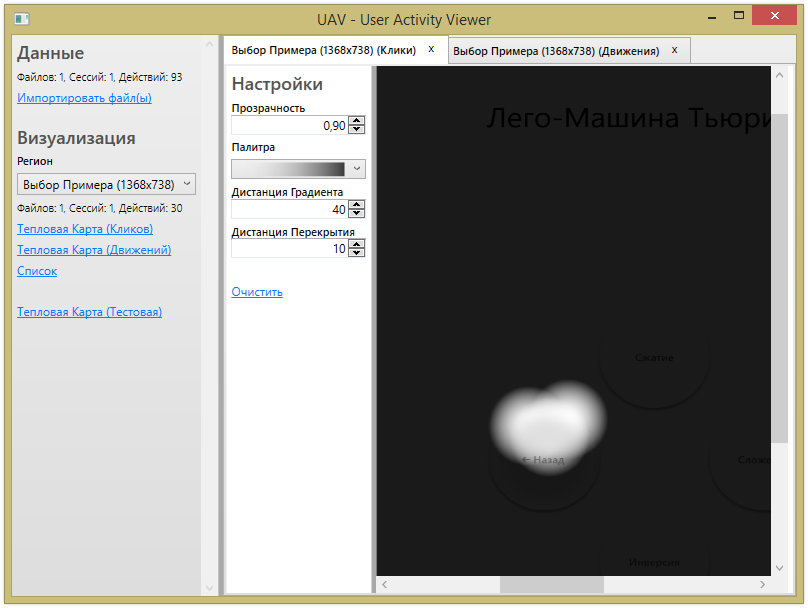


Рисунок 3.2.4 – Карта прозрачности в UAV (клики мышью)

Построение тепловых карт осуществляется при помощи алгоритма, описанного во второй главе. Интенсивность при этом указывается через канал прозрачности (англ. Alpha Channel) в цветовой схеме RGBA, который суммируется образуя совокупную интенсивность. Цвет точки определяется при помощи пиксельного шейдера (рисунок 3.2.5), который на основании совокупной интенсивности извлекает нужный цвет из используемой палитры.

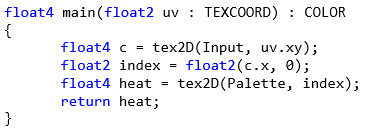


Рисунок 3.2.5 – Пиксельный шейдер для тепловой карты

Кроме тепловых карт, поддерживается отображение данных активности пользователя в виде списка (таблицы) для выбранного региона (рисунок 3.2.6).

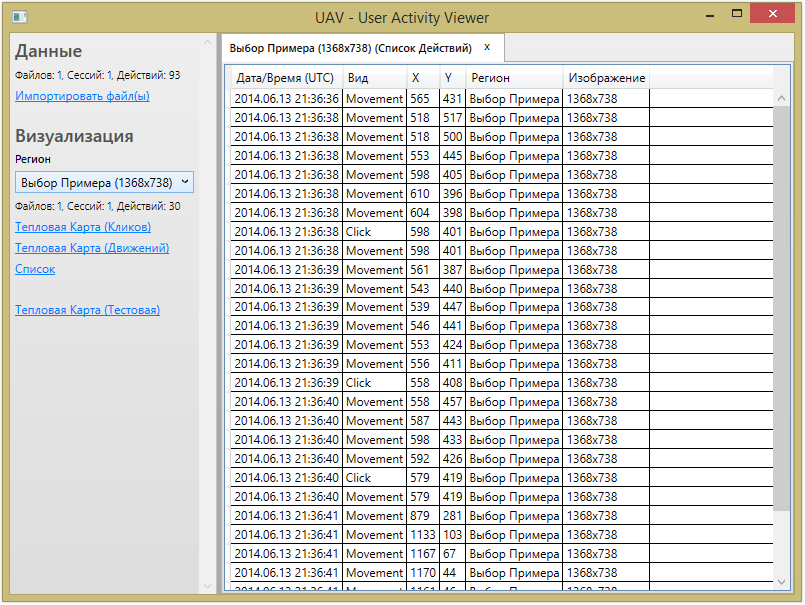


Рисунок 3.2.6 – Список активностей (действий) в UAV

## 3.3. Тестирование

Тестирование является обязательной частью процесса разработки ПО. Так как основная часть разработки велась в соответствии с принципами BDD, то фактически создание модульных автоматических тестов производилось параллельно процессу реализации требуемого функционала. Тесты создавались при этом по принципу «Черного Ящика» (англ. Black Box), когда конкретная реализация считается скрытой и тестируется только внешнее поведение методов. Например, на рисунке 3.3.1 представлен модульный автоматический тест для проверки определения региона по дочернему элементу управления.

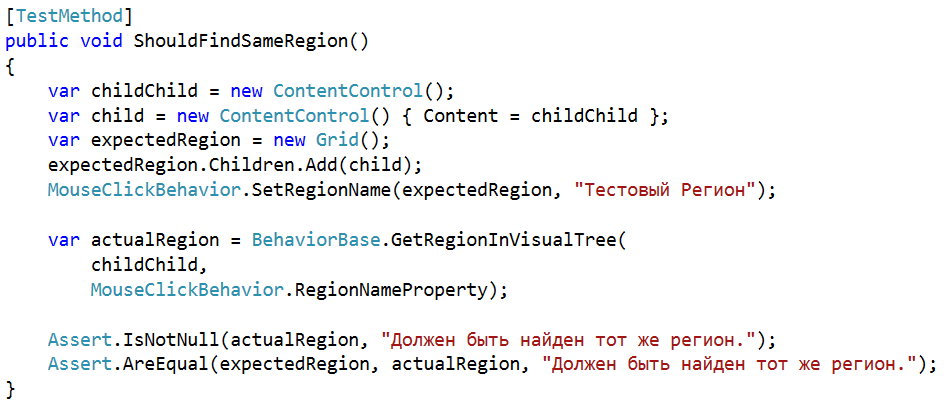


Рисунок 3.3.1 – Модульный автоматический тест для проверки определения региона по дочернему элементу управления

После реализации и тестирования модулей по отдельности, производилось их совместное (системное, англ. System, End-To-End) тестирование сценарным методом. Подключение и апробация UACL-модуля при этом производились на основе настольного WPF-приложения «Лего-Машина Тьюринга», разработанного в рамках научно-практического проекта кафедры «Прикладная информатика и программная инженерия». Тестировался следующий полный сценарий сбора, загрузки и последующей визуализации данных активности пользователя.

* Запуск приложения «Лего-Машина Тьюринга», с подключенным UACL-модулем. Проверяется корректность начала работы приложения при подключенном к нему UACL-модуля.
* Работа (взаимодействие) с приложением «Лего-Машина Тьюринга» и его закрытие. Проверяется корректность завершения работы приложения при подключенном к нему UACL-модулем. Проверяется создание UAD-файла со всеми собранными данными об активности.
* Копирование UAD-файлов. Проверяется что файлы не должны быть заняты другим процессом и должны быть доступны для копирования.
* Запуск UAV и загрузка UAD-файлов. Проверяется корректность начала работы UAV. Проверяется корректность загрузки UAD-файлов и отображение количества загруженных файлов и состав данных в них.
* Выбор региона в соответствующем выпадающем списке. Проверяется активность кнопок для визуализации после выбора региона. Проверяется корректное отображение состава загруженных данных конкретно для выбранного региона.
* Открытие, настройка и закрытие тепловой карты кликов для выбранного региона. Проверяется возможность настройки параметров визуализации и корректность перестроения тепловой карты в соответствии с текущими настройками. Проверяется скрытие вкладки из рабочей области при закрытии тепловой карты.
* Открытие, настройка и закрытие тепловой карты движений для выбранного региона. Проверяется возможность настройки параметров визуализации и корректность перестроения тепловой карты в соответствии с текущими настройками. Проверяется скрытие вкладки из рабочей области при закрытии тепловой карты.
* Открытие и закрытие списка (таблицы) для выбранного региона. Проверяется скрытие вкладки из рабочей области при закрытии тепловой карты.

Кроме сценарного тестирования, было решено провести тестирования основанное на рисках (англ. Risk Based). Было выделено несколько наиболее критичных рисков.

* Ошибка доступа к файловой системе при создании UAD-файла UACL-модулем по завершению работы анализируемого приложения.

Способ тестирования: Запуск анализируемого приложения в условиях отсутствия прав в файловой системе на создание новых файлов.

Результат тестирования: UAD-файл не был создан, работа анализируемого приложения не была нарушена.

* Некорректный формат загружаемого UAD-файла при работе с UAV.

Способ тестирования: Внесение ошибок в структуру XML-содержимого UAD-файла, запуск UAV, загрузка UAD-файла.

Результат тестирования: UAD-файл не загружен, работа UAV не нарушена.

* Ошибка доступа к файловой системе при загрузке изображения региона при открытии тепловой карты в UAV.

Способ тестирования: Запуск UAV в условиях отсутствия прав на чтение файлов в файловой системе.

Результат тестирования: изображение не загружено, работа UAV не нарушена.

Тестирование производилось на нескольких конфигурациях персональных компьютеров (ПК) и сред выполнения:

* Windows 7 Starter (32-bit), .NET Framework 4.0,   
  Inter(R) Core(TM) i3-2100 CPU @ 2.60 GHz (x86), 2 Gb RAM;
* Windows 8.1 Single Language (64-bit), .NET Framework 4.0,   
  Inter(R) Core(TM) i5-3317U CPU @ 1.70 GHz(x64), 6 Gb RAM.

Все указанные варианты тестирования были успешно пройдены разработанным ПО, на основании чего его можно считать готовым к дальнейшей эксплуатации.

# 4. Реализация, проверка и апробация разработанных математических моделей и методов

В четвертой главе осуществляется реализация, проверка и апробация разработанных математических моделей и методов, путем построения карт активности и расчета временных на основе реальных собранных данные, которые будут расписаны в автореферате позднее.

## 4.1. Наполнение онтологии экземплярами

Для качественного анализа и проверки адекватности предложенной онтологии требуются достаточно большие наборы данных, собранные для отдельных типов интерфейсов. Предполагается, что данные в общем случае, будут собираться автоматически, то есть необходимы инструменты (программное обеспечение) для автоматического наполнения онтологии экземплярами на основе данных активности пользователей приложений с различными типами интерфейсов. При этом формат и способ хранения данных могут зависеть от конкретной реализации. Главное, чтобы данные содержали в себе всю информацию соответственно структуре классов представленной онтологии и были доступны для последующего преобразования в онтологический вид.

Авторами предлагается использовать RDF-модель (Resource Description Framework) в стандартной RDF/XML нотации для упрощения взаимодействия различного программного обеспечения при сборе, передаче и анализе данных активности пользователей.

В рамках данной работы эксперимент по проверки адекватности онтологии был проведен для прикладного программного обеспечения, относящегося к категории настольных приложений для операционных систем семейства Windows. Разработанное авторами программное обеспечение [15] позволяет собирать базовые данные о действиях пользователя, взаимодействующего с таким типом интерфейсов, а именно: передвижение курсора мыши, одинарный клик мыши, вызванная команда и т.п. Собранные данные могут сохраняться в RDF/XML формате, соответствующем разработанной онтологии. В рамках эксперимента данные собирались для 6 программных приложений, в том числе упомянутого выше комплекса «Лего-машина Тьюринга». После этого данные из различных источников были использованы для построения тепловых карт отдельных регионов приложений.

Например, приведенная выше тепловая карта (рис. 6) построена на основе собранных таким образом данных, фрагмент которых выглядит следующим образом:

<rdf:RDF

xmlns="http://ontologies/usability#"

xml:base="http://ontologies/usability"

xmlns:us="http://ontologies/usability#"

xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">

<Session

us:hasStartDateTime="2017-10-12T13:18:58+04:00"

us:hasEndDateTime="2017-10-12T13:45:51+04:00"

us:hasUid="ca0ad458-d3ab-4cd3-8192-5a29511b0f68">

<wasPerformedBy>

<User us:hasName="Школьник" />

</wasPerformedBy>

<contains rdf:parseType="Collection">

<Region us:hasName="Пример: Сложение кубиков">

<contains rdf:parseType="Collection">

<Variation>

<contains rdf:parseType="Collection">

<SingleClickMouseEvent

us:hasDateTime="2017-10-12T13:20:00+04:00"

us:hasInRegionX="27"

us:hasInRegionY="627" />

<SingleClickMouseEvent

us:hasDateTime="2017-10-12T13:20:04+04:00"

us:hasInRegionX="28"

us:hasInRegionY="625" />

...

</contains>

</Variation>

</contains>

</Region>

</contains>

</Session>

</rdf:RDF>

Пример данных содержит одну сессию с датой начала 2017-10-12T13:18:58+04:00, датой окончания 2017-10-12T13:45:51+04:00 и уникальным идентификатором «ca0ad458-d3ab-4cd3-8192-5a29511b0f68». Сессия исполнена пользователем с именем «Школьник» и содержит один регион с именем «Пример: Сложение кубиков» с одной вариацией по-умолчанию без имени. Вариация содержит два события типа «SingleClickMouseEvent» (одинарный клик мыши), прочие событие опущены для краткости примера.

Согласно экспертной оценке, построенные на основе онтологических данных тепловые карты могут эффективно использоваться при анализе удобства использования программного обеспечения.

Таким образом, показано, что предложенная онтология адекватна структуре данных активностей пользователей прикладного программного обеспечения, относящегося к категории настольных приложений для операционных систем семейства Windows.

## 4.2. Построение тепловой карты

Полученные данные позволили построить тепловые карты регионов и провести анализ удобства использования. Например, тепловая карта, представленная на рисунке 6, демонстрирует множественные попытки взаимодействия пользователей с полосой прокрутки, что является индикатором ошибки для данного региона, т.к. пользователь пытается взаимодействовать с неактивным элементом интерфейса. В то же время, отсутствие «промахов» пользователями мимо кнопок означает отсутствие ошибок в основном сценарии взаимодействия с программным интерфейсом.

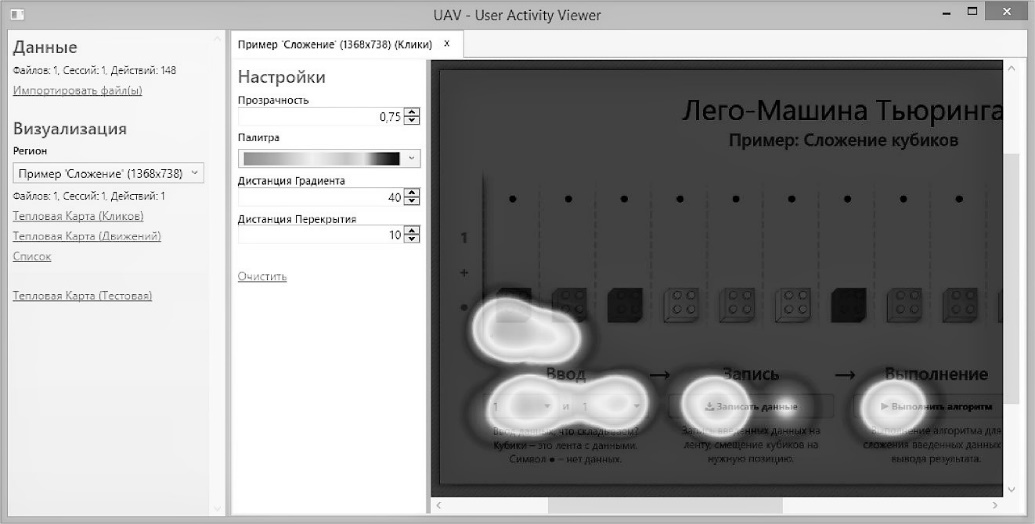


Рис. 6. Тепловая карта одного из регионов комплекса «Лего-машина Тьюринга»

Осуществляется реализация, проверка и апробация разработанных математических моделей и методов, путем построения карт активности и расчета временных на основе реальных собранных данные, которые будут расписаны в автореферате позднее.

Существующие визуализации активности пользователей методы не содержат параметра для управления дистанцией перекрытия, считая перекрывающими точки только при полном совпадении координат, что приводит к резкому росту площади высокой интенсивности и ухудшает наглядность. С другой стороны, они используют технику альфа-смешения (англ. alpha blending) для расчета результирующего цвета (интенсивности), что наоборот уменьшает реальное значение интенсивности.

Для апробации метода визуализации необходимо проверить различные случаи. Ниже приведены примеры построены тепловых карт, сверху-вниз слева направо:

* r=3, d=0, int=1, method=alpha;
* r=80, d=0, int=1, method=alpha;
* r=80, d=9, int=0.5, method=alpha;
* r=80, d=10, int=0.33, method=alpha;
* r=80, d=10, int=0.33, method=**sum**;
* r=80, d=40, int=0.2, method=**sum**;

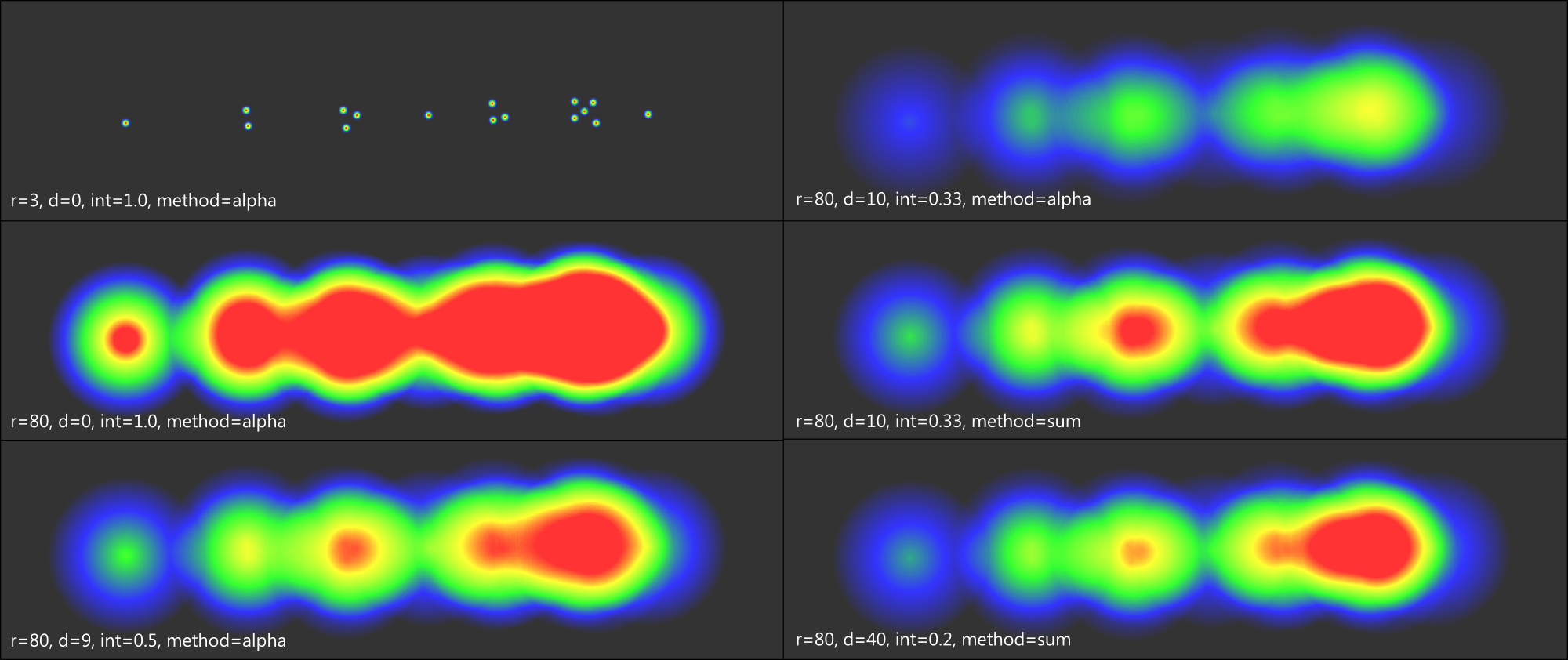


Рис. 7. Примеры построения тепловых карт

# ЗАКЛЮЧЕНИЕ

В процессе научно-практических исследовательских работ, были поставлены и успешно решены следующие задачи:

1. Исследованы существующие математические модели и методы применяемые при оценке удобства использования программного обеспечения.

2. Разработана математическая модель предметной области «Удобство использования ПО», предназначенную для описания данных активности пользователей.

3. Разработана математическая модель активности пользователей, предназначенную для визуализации данных активности.

4. Разработан математической метод построения карт активности пользователей в виде тепловых карт на основе данных активности.

5. Разработан математический метод поиска повторяющихся шаблонов событий, порожденных активности пользователей.

6. Создан комплекс проблемно-ориентированных программ, реализующий разработанные математические модели и методы, позволяющий собирать и анализировать данные активности пользователей.

7. Проведена апробация разработанных математических моделей и методов применительно к задачам анализа удобства использования прикладного программного обеспечения для операционных систем семейства Windows.

При решении первой задачи была изучена история развития научно-прикладной дисциплины юзабилити, проанализированы современные методы оценки юзабилити пользовательских интерфейсов и применяемые инструменты. Было выявлено два важных момента. Во-первых, главным объектом исследования во всех методах при оценке юзабилити является процесс взаимодействия пользователя с компьютерной системой. Во-вторых, несмотря на многообразие методов и инструментов оценки юзабилити, большинство из них ориентировано исключительно на веб-приложения.

При решении второй задачи было исследовано понятие активности пользователя, проанализированы способы визуализации данных активности пользователя и определено, что наибольшей популярностью пользуется способ визуализации, называемый «тепловая карта», так как он обеспечивает наибольшую наглядность. Как наиболее приоритетные для последующего анализа и применения виды активности пользователя, были выбраны клики (нажатия левой кнопки) мышью и движения курсора. Однако, в открытых научных источниках не удалось обнаружить описанных алгоритмов, адаптированных для визуализации таких видов активности. Различные программные продукты, в которых реализованы свои собственные алгоритмы, являются закрытыми.

Поэтому, для выполнения третьей задачи, был разработан алгоритм визуализации точечных данных активности пользователя, оптимизированный для отображения больших объемов данных, с большой плотностью расположения точек кликов и движений мышью.

Было спроектировано программное обеспечение для сбора и визуализации данных активности пользователя настольных приложений любых типов и основанных на различных платформах. Его структура состоит из двух уровней и схемы для хранения данных в виде XML-файлов и изображений регионов. На первом уровне располагаются специальные модули для отслеживания и сбора данных активности пользователя. Они подключаются в настольные приложения определенным образом, в зависимости от базовой платформы (.NET, Java, Delphi). После подключения и настройки интересующих для анализа регионов приложения, модули начинают отслеживать действия пользователя и записывать собранные данные в виде XML-файлов соответствующих специальной XSD-схеме. Второй уровень представляет собой настольное приложение для загрузки этих файлов и визуализации определенным образом данных хранящихся в них. В нем реализуется алгоритм для построения тепловых карт, составленный в рамках решения третьей задачи.

Таким образом, поставленные задачи были решены, цель работы достигнута. По основным результатам работы сделано 9 докладов на 9 международных, всероссийских и региональных конференциях (Саратов, Уфа, Житомир (Украина), Лорман (США)). Имеются следующие публикации по исследуемой научной тематике в периодических изданиях и материалах конференций: [40], [41], [42], [43] и [44].

Полученные результаты, в дальнейшем будут использованы в последующих научных исследованиях при разработке методов и алгоритмов оценки юзабилити настольных приложений на основании данных активности пользователя. В дальнейшем планируется что спроектированное ПО, на основании загруженных данных и специальных алгоритмов, сможет производить автоматический анализ и давать оценку юзабилити для настольных приложений целиком или их отдельных регионов.

# СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Navigli R., Velardi P. Semantic interpretation of terminological strings // Proc. 6th Int’l Conf. Terminology and Knowledge Eng. 2002. P. 95–100.

2. Frantzi, K. Automatic recognition of multi-word terms: The C-value/NC-value method / K. Frantzi, S. Ananiadou, H. Mima // International Journal on Digital Libraries. — 2000. — Т. 3, No 2. — С. 115—130.

3. Evans D. A., Lefferts R. G. CLARIT-TREC experiments // Information processing & management. 1995. Vol. 31, no. 3. P. 385–395.

4. Baroni, M. BootCaT: Bootstrapping corpora and terms from the web / M. Baroni, S. Bernardini // Proceedings of LREC. — 2004. — Т. 4. — С. 1313—1316. — URL: http://www.cs.utah.edu/nlp/readinglist/BaroniB04.pdf.

5. Sclano, F. TermExtractor: a Web Application to Learn the Common Terminology of Interest Groups and Research Communities / F. Sclano, P. Velardi // Proceedings of the 9th Conference on Terminology and Artificial Intelligence (TIA 2007). — 2007.

6. Park, Y. Automatic glossary extraction / Y. Park, R. J. Byrd, B. K. Boguraev // Proceedings of the 19th international conference on Computational linguistics -. — 2002. — Т. 1. — С. 1—7. — URL: http://dl.acm.org/citation.cfm?id=1072228.1072370.

7. University of surrey participation in trec8: Weirdness indexing for logical document extrapolation and retrieval (wilder) / K. Ahmad, L. Gillam, L. Tostevin et al. // The Eighth Text REtrieval Conference (TREC-8). 1999.

8. Meijer Kevin, Frasincar Flavius, Hogenboom Frederik. A semantic approach for extracting domain taxonomies from text // Decision Support Systems. 2014. Т. 62. С. 78–93.

9. Браславский П.И., Соколов Е.А. Автоматическое извлечение терминологии с использованием поисковых машин Интернета // Компьютерная лингвистика и интеллектуальные технологии. Тр. Международной конференции «Диалог». 2007. С. 89-94.

10. Голомазов Д. Д. Методы и средства управления научной информацией с использованием онтологий // Диссертация кандидата физико-математических наук. Москва. 2012.

11. Астраханцев, Н. А. Методы и программные средства извлечения терминов из коллекции текстовых документов предметной области // Диссертация кандидата физико-математических наук. Москва. 2014.

12. Vivaldi J., Rodrı́guez H. Using Wikipedia for term extraction in the biomedical domain: first experiences // Procesamiento del Lenguaje Natural. 2010. Vol. 45. P. 251–254.

13. Korobov, M. Morphological Analyzer and Generator for Russian and Ukrainian Languages / M. Korobov // Analysis of Images, Social Networks and Texts. Т. 542 / под ред. M. Y. Khachay [и др.]. — Springer International Publishing, 2015. — С. 320—332. — (Communications in Computer and Information Science). — URL: http://dx.doi.org/10.1007/978-3-319-26123-2\_31.

\