Онтология предметной области «Удобство использования программного обеспечения»

А.А. Сытник <as@sstu.ru>
Т.Э. Шульга <shulga@sstu.ru>
Н.А. Данилов <Nikita_Danilov@outlook.com>
Саратовский государственный технический университет имени Ю. А. Гагарина,
410054, г.Саратов, ул.Политехническая, д. 77

Аннотация. В статье представлена онтология предметной области «Удобство использования программного обеспечения». Описываются преимущества, которые может дать ее использование при анализе и оценке удобства использования программных продуктов. Представлены диаграмма классов предлагаемой онтологии и текстовое описание входящих в ее состав классов, объектных свойств и свойств данных. Приводятся примеры вопросов, на которые может отвечать онтология. Предлагается классификация возможных методик анализа и оценки удобства использования на основе представленной онтологии и описываются некоторые из них.

Ключевые слова: удобство использования программного обеспечения; пользовательский интерфейс; онтология; онтологический инжиниринг

DOI: 10.15514/ISPRAS-2018-30(2)-10

Для цитирования: Сытник А.А., Шульга Т.Э., Данилов Н.А. Онтология предметной области «Удобство использования программного обеспечения». Труды ИСП РАН, том 30, вып. 2, 2018 г., стр. 195-214. DOI: 10.15514/ISPRAS-2018-30(2)-10

1. Введение

Стандарты ГОСТ Р ИСО 9241-210–2016 [1] и ISO 9241-11:1998 [2] определяют пригодность использования как свойство системы, продукции или услуги, при наличии которого установленный пользователь может применить продукцию в определенных условиях использования для достижения установленных целей с необходимой результативностью, эффективностью и удовлетворенностью. В профессиональной среде программистов более устоявшимся термином для данного свойства программного продукта является термин юзабилити (usability). Уровень удобства использования программного интерфейса влияет на качество всего программного обеспечения в целом [3]. Заметим, что в

российских стандартах качества программных средств термин «пригодность использования» упоминается как «удобство использования» и «практичность».

Существуют различные подходы и методики, применяемые для анализа удобства использования [4]. Однако, большинство из них основывается на эвристическом подходе, то есть на заранее собранных рекомендациях и гипотезах относительно того, как пользователь взаимодействует с программным интерфейсом системы. По этой причине их нельзя назвать в достаточной мере формализованными и их эффективность во многом зависит от уровня эксперта. Отсутствует и общая модель хранения данных активности пользователя, которая бы позволила свободно обмениваться данными, собираемыми в процессе тестирования удобства использования, и применять эти данные для ее анализа.

Возможным решением указанных проблем является разработка модели предметной области удобства использования в виде онтологии. Тенденции развития семантического веба позволяют утверждать, что онтология, как модель представления знаний, имеет ряд преимуществ. Согласно определению консорциума W3C, под онтологией понимается формальная модель представления знаний в некоторой предметной области, описывающая типы объектов (классы), взаимосвязи между ними (свойства), и способы совместного использования классов и свойств (аксиомы) [5].

Онтологии, публикуемые в вебе в стандартных форматах, позволяют упростить процесс распространения знаний и их повторного использования. Кроме того, модель онтологии подразумевает возможность ее последующего расширения или уточнения с целью использования в любых программных приложениях определенной предметной области.

Заметим, что в последние годы большое количество научных работ посвящено проблемам онтологического моделирования в разных предметных областях (например, [6-9]). В данной работе описывается онтология предметной области «Удобство использования программного обеспечения» и приводится описание вариантов ее возможного использования в методиках анализа юзабилити. При этом из трех групп метрик оценки удобства использования, определенных в стандартах [1,2] − результативность, эффективность, удовлетворенность, − в работе описываются методики оценки удобства использования, ориентированные на эффективность, а именно, на время выполнения задач пользователя.

2. Обзор онтологий

Основным объектом исследования при оценке удобства использования являются действия пользователя [1], процесс его работы с программной системой, то есть его активность: движение курсора мыши, клики клавиш мыши, нажатие клавиш клавиатуры, различные формы взаимодействия с сенсорным экраном и т.п. Таким образом, для оценки удобства использования

необходимо аккумулировать данные активности пользователей с их привязкой к пользовательскому интерфейсу.

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.

Энн Блэндфорд (Ann Blandford) и Томас Грин (Thomas Green) еще в 1997 году предложили онтологическую модель построения эскизов (Ontological Sketch Model, 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 (entities), действия (actions), взаимосвязи (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 (attributes) – дополнительные характеристики которыми обладает сущность, например, символ имеет шрифт, размер;
- доступность (accessibility) – на уровне понимания пользователя, уровне устройства или системы и общие;
- релевантность (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 (persistent visibility) – может ли пользователь увидеть сущность, существует она условно или прозрачна;
- маскировка (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 (effect), получаемый после совершения действия;
- контекст (context), контекстная информация.

Взаимосвязь – связь между сущностями, обладает свойствами:

- тип (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, так как основная идея ее 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.

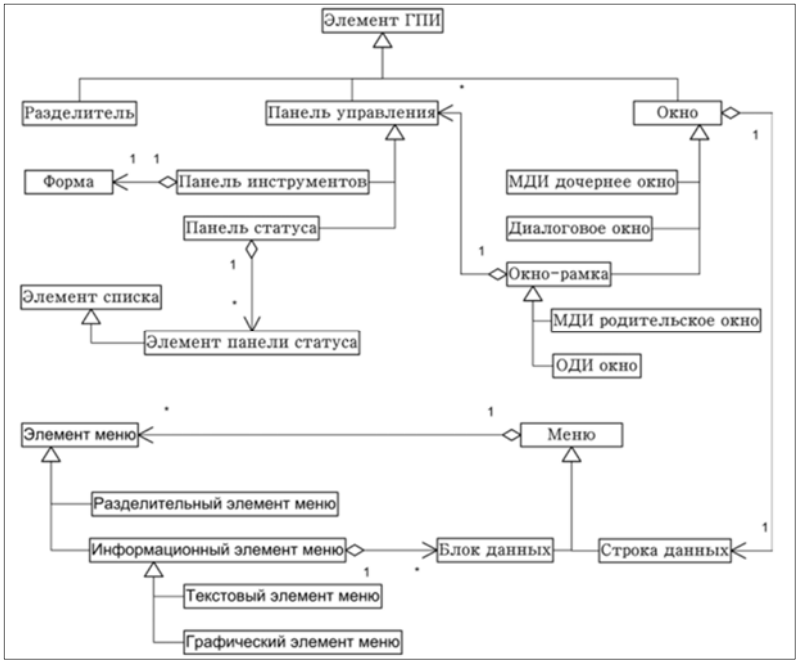

Рис. 1. Фрагмент иерархии элементов ГПИ
Fig. 1. A fragment of the GUI elements hierarchy

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» (user interface, UI) [9] под авторством Тимоти Джона Бернерс-Ли (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.

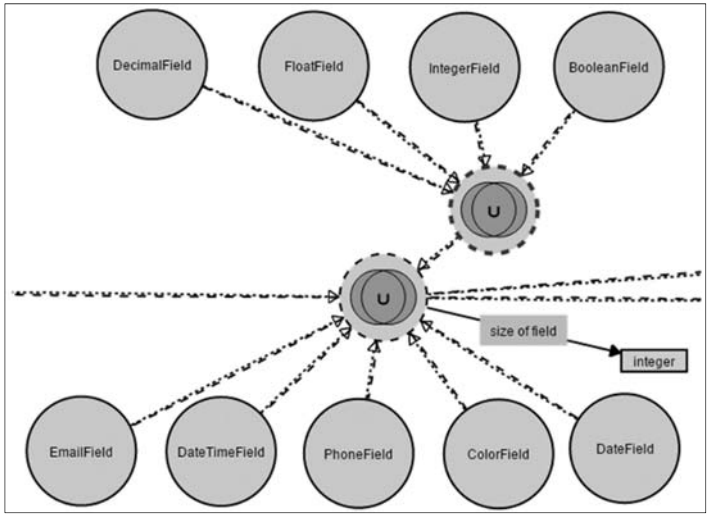

Рис. 2. Фрагмент онтологии пользовательского интерфейса
Fig. 2. A fragment of the User Interface ontology

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 удается 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».

3. Описание онтологии предметной области «Удобство использования»

Онтология предметной области «Удобство использования программного обеспечения» в первую очередь предназначена для фиксации данных о взаимодействии пользователя с программной системой посредством графического пользовательского интерфейса. При оценке удобства использования необходимы данные именно об активности пользователя. Учитывая современное развитие технологий, можно выделить огромное количество способов и видов взаимодействия пользователя с пользовательским интерфейсом.

Подробное описание пользовательского интерфейса, в свою очередь, с детализацией всех элементов по их типам и характеристикам в данный момент, по мнению авторов, не представляется необходимым. Для решения базовых задач оценки удобства использования достаточно общего разбиения пользовательского интерфейса на интересующие эксперта участки (регионы). Таким образом, предметная область «пользовательский интерфейс» не покрывается данной онтологией.

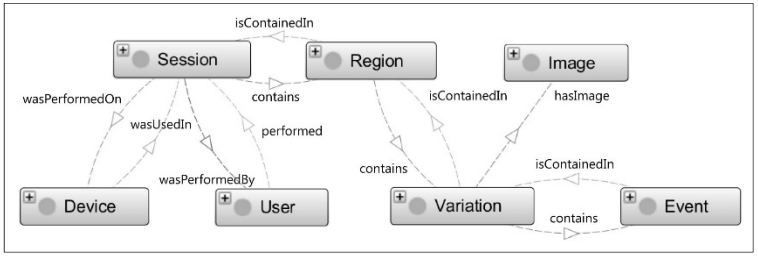

Рис. 3. Фрагмент структуры классов онтологии «Удобство использования программного обеспечения»
Fig. 3. Classes structure fragment of the "Software Usability" ontology

Опишем основные понятия, используемые в рассматриваемой предметной области и взаимосвязи между ними, т.е. классы и свойства предлагаемой

онтологии (рис. 3). Заметим, что на рис. 3 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.

Класс Session предназначен для описания сессии − временного промежутка, в течение которого пользователь взаимодействует с программной системой. В рамках сессии накапливаются и хранятся все данные об активности пользователя, события, действия пользователя и вспомогательная информация. Свойства данных hasStartDateTime и hasEndDateTime позволяют указать границы промежутка времени.

Класс Device описывает устройство, на котором выполнялась сессия. Device в текущей реализации являются простым классом, содержащим лишь базовую информацию для понимания «где» выполнялась сессия работы с программной системой. Экземпляры устройств привязываются к сессиям свойством wasUsedIn, которое является инверсивным по отношению к wasPerformedOn. Свойство данных hasName позволяет указать краткое имя для устройства.

Класс User описывает пользователя, которым выполнялась сессия. User в текущей реализации являются простым классом, содержащим лишь базовую информацию для понимания «кем» выполнялась сессия работы с программной системой. Экземпляры пользователей привязываются к сессиям свойством performed, инверсивным по отношению к wasPerformedBy. Свойство данных hasName позволяет указать краткое имя пользователя.

Для каждого экземпляра сессии, должен быть указан строго один экземпляр устройства, на котором она совершалась (объектное свойство wasPerformedOn), и строго один экземпляр пользователя, которым она совершалась (объектное свойство wasPerformedBy).

Класс Region описывает регион, т.е. область пользовательского интерфейса, например, окно целиком, либо его отдельную часть. Регион связывается с сессией транзитивным свойством isContainedIn. Экземпляры сессий связываются с коллекцией экземпляров регионов с помощью объектного транзитивного свойства contains. Свойство данных hasName позволяет указать имя региона.

Класс Variation предназначен для описания вариации региона, так как каждый регион, в свою очередь, может иметь одну или более вариаций. Под вариацией региона в общем случае может пониматься уникальное сочетание параметров высоты и ширины изображения региона, либо их диапазон. Выделение вариаций необходимо при анализе адаптивного пользовательского интерфейса, когда внешний вид региона и, возможно, функциональность, меняется в зависимости от его размеров. Например, крупные кнопки с подробными надписями могут заменяться на более мелкие с пиктограммами и без надписей (рис. 4). Минимальные и максимальные границы ширины и высоты определяются свойствами данных hasMinWidth, hasMaxWidth, hasMinHeight,

hasMaxHeight. Свойство данных hasName определяет название вариации. Вариация связывается с регионом свойством isContainedIn. Регион связывается со всеми своими вариациями объектным свойством contains.

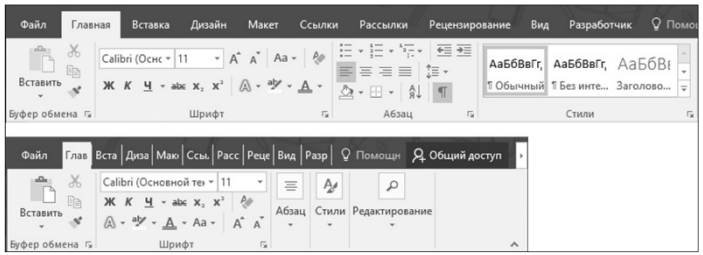

Рис. 4. Пример вариаций региона панели инструментов текстового процессора MS Word
Fig. 4. Region variations example for the MS Word toolbar

Класс Image используется для описания изображения. При оценке удобства использования и анализе данных активности пользователя может потребоваться знание того, как выглядел пользовательский интерфейс при работе с ним, т.е. его изображение. Например, кроссплатформенное прикладное программное обеспечение работает в самых разных условиях и на разных операционных системах. Общепринятой практикой является что приложение «подстраивается» под условия работы.

Для каждого изображения создается экземпляр класса Image, который связывается строго с одним экземпляром вариации. Свойства данных hasWidth и hasHeight позволяют определить ширину и высоту изображения. Свойства данных hasDpiX и hasDpiY определяют разрешение изображения. Объектное свойство hasImage описывает прямую связь вариации с изображением, а инверсивное ему объектное свойство wasImaged описывает обратную.

Класс Event представляет собой базовый класс для событий, которые могут происходить при взаимодействии пользователя с программной системой. Вариация имеет связь contains со всеми событиями. Объектное свойство isContainedIn позволяет указать строго одну вариацию в которой произошло событие.

Событием может быть любое действие пользователя, команда и т.д. Набор информации о событии зависит от его конкретного типа. Общим для всех событий является свойство данных hasDateTime, которое указывает, когда событие произошло. Класс Event можно считать абстрактным, т.к. он не несет информации о типе события. Для анализа требуется работать с детализированными типами событий, подклассами класса Event. Опишем структуру подклассов событий (рис. 5).

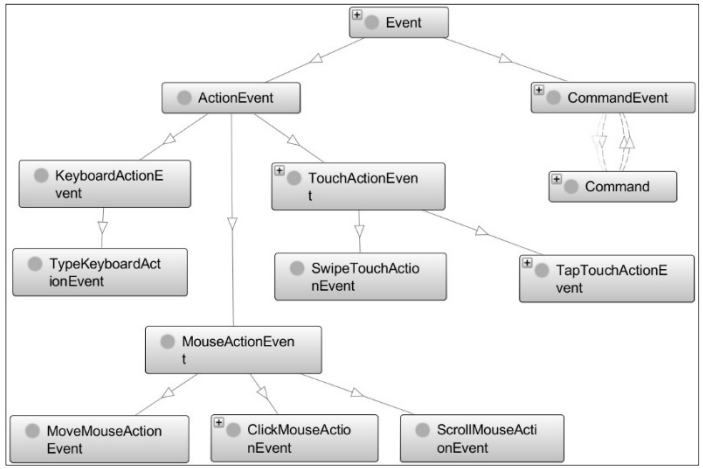

Рис. 5. Подклассы класса Event онтологии «Удобство использования программного обеспечения»
Fig. 5. Event subclasses of the "Software Usability" ontology

Класс ActionEvent является подклассом Event и является надклассом всех классов событий действий. Они служат для фиксации действий пользователя, которые относятся к механическому взаимодействию пользователя с программным интерфейсом (движение курсора мыши, клики), обычно, не несущим связи с функциональностью программного продукта и функциональными требованиями напрямую.

Подкласс KeyboardActionEvent описывает класс для событий действий, связанных с манипуляцией пользователя с клавиатурой.

Подкласс TypeKeyboardActionEvent описывает событие нажатие клавиши клавиатуры.

Подкласс MouseActionEvent описывает события действий, связанных с манипуляцией пользователя с мышью.

Подкласс ClickMouseActionEvent соответствуют событию нажатия клавиши мыши. Для события необходимо указание информации о координатах курсора мыши, где произошло событие, с помощью свойств hasInRegionX и hasInRegionY. Подклассы SingleClickMouseEvent и DoubleClickMouseEvent описывают одинарный и двойной клики соответственно.

Подкласс MoveMouseActionEvent соответствует событию движения курсора мыши. Свойства hasInRegionX и hasInRegionY позволяют указать новые координаты курсора мыши после перемещения.

Подкласс TouchActionEvent описывает события действий, связанных с взаимодействием пользователя с сенсорным экраном (тачскрин, англ. touchscreen), например, дисплеем телефона или планшетного компьютера.

Подкласс TapTouchActionEvent описывает касание пользователем сенсорного экрана. Можно считать условным аналогичным ClickMouseActionEvent, но для сенсорного экрана.

Подклассы SingleTapTouchActionEvent, DoubleTapTouchActionEvent, LongTapTouchActionEvent, HoldTapTouchActionEvent описывают соответственно: одиночное быстрое касание, двойное быстрое касание, одинарное долгое касание и касание с удержанием. Подобная детализация важна в случае сенсорных экранов, поскольку способ касания определяет ответную реакцию программного обеспечения.

Подкласс SwipeTouchActionEvent описывает событие способа взаимодействия пользователя с интерфейсом, когда пользователь проводит пальцем или пальцами в одном направлении.

Класс CommandEvent является подклассом класса Event и описывает события команд, или командные события. Они служат для фиксации факта вызова определенной команды (функции) пользователем.

Класс Command описывает команду, которая является функциональным действием пользователя (обычно, связанным с функциональными требованиями), подразумевающим ответную реакцию программной системы. Свойство hasName позволяет указать название команды.

Все командные события связываются с конкретными экземплярами команд свойством wasAssociatedWith, а команды с командными событиями, в свою очередь, свойством wasInvokedIn. Для упрощения последующего использования онтологии, данная связь дополнительно дублируется транзитивными свойствами contains и isContainedIn, где событие содержит команду, а команда содержится в событии.

4. Использование онтологии при анализе и оценке удобства использования

Предложенная онтология может быть использована при анализе и оценке удобства использования пользовательских интерфейсов программного обеспечения. Для этого могут потребоваться различные выборки данных, связанные с различными событиями. Важно отметить, что не для каждого выполненного действия гарантируется исполнение команды. Перемещение курсора, в большинстве случаев, не вызывает выполнение какой-либо команды. Клик курсора мыши в районе кнопки может не привести к выполнению команды, если кнопка была неактивна или если пользователь «промахнулся». Однако, независимо от этого любое механическое взаимодействие пользователя с интерфейсом напрямую затрачивает его ресурсы, такие как время и внимание.

Прежде всего, приведем примеры вопросов, на которые может отвечать описанная онтология.

- Какие команды совершены за сессию?

- Какие действия совершены в регионе?
- Сколько кликов мышью совершено в регионе?
- Какие команды совершены пользователями в регионе?
- Какие вариации региона встречаются в сессии?

Допустим, существует регион с названием «SomeRegion» (свойство данных hasName). Приведем пример SPARQL-запроса, отвечающего на вопрос «Какие команды совершены в регионе 'SomeRegion'?»:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX usa:
<http://www.semanticweb.org/nikita_danilov/ontologies/usability#>
SELECT DISTINCT ?cname
WHERE {
    ?r rdf:type usa:Region .
    ?r usa:hasName ?rname .
    ?r usa:hasName
"SomeRegion"^^<http://www.w3.org/2001/XMLSchema#string> .
    ?v rdf:type usa:Variation .
    ?ce rdf:type usa:CommandEvent .
    ?c rdf:type usa:Command .
    ?c usa:hasName ?cname .
    ?r usa:contains+ ?c }
```

Транзитивность свойства contains позволяет избежать перечисления полной вложенности сущностей (вариации в регионе, командное событие в вариации, команда в командном событии).

Если требуется подсчитать в целом по всем имеющимся данным сколько раз была вызвана каждая команда в каждом регионе, то SPARQL-запрос будет выглядеть следующим образом:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX usa:
<http://www.semanticweb.org/nikita_danilov/ontologies/usability#>
SELECT ?rname ?cname (count(?c) as ?count)
WHERE {
    ?r rdf:type usa:Region .
    ?r usa:hasName ?rname .
    ?v rdf:type usa:Variation .
    ?ce rdf:type usa:CommandEvent .
    ?c rdf:type usa:Command .
    ?c usa:hasName ?cname .
    ?r usa:contains+ ?c }
GROUP BY ?rname ?cname
ORDER BY ?rname ?cname
```

Подобные запросы могут использоваться в составе методик анализа и оценки удобства использования. Авторами предлагается их классификация в зависимости от базовой метрики удобства использования [1], на которую они ориентированы. Всего стандарт ISO 9241-11:1998 [2] определяет 3 группы таких метрик (показателей).

- Результативность (англ. effectiveness) – степень реализации

запланированной деятельности и достижения запланированных результатов.

- Эффективность (англ. efficiency) – связь между достигнутым результатом и использованными ресурсами.
- Удовлетворенность (англ. satisfaction) – комфорт и приемлемость использования, отсутствие дискомфорта при использовании продукции или положительное отношение к ней.

Результативность и эффективность основаны на достаточно формальных показателях, таких как количество выполненных задач и объем затраченных ресурсов. Поэтому, по мнению авторов, данные метрики возможно формализовать с достаточной степенью точности.

Результативность – точность и полнота, с которой пользователи достигают поставленных целей, успешность выполнения промежуточных задач необходимых для достижения цели. Ориентированные на результативность (effectiveness-oriented) методики могут быть связаны непосредственно показателями с успешности выполнения задач. Существуют различные показатели результативности. Стандарт ISO 9241-11:1998 [2] определяет, например, такие как: процент достигнутых целей, процент пользователей, успешно завершивших задачу, средняя точность выполненных задач.

Важно отметить, что понятие «задача» в предметной области «Удобство использования программного обеспечения» является комплексным и вариативным. В общем виде, под задачей понимается деятельность, необходимая пользователю для достижения определенной поставленной цели [1]. Однако, критерии определения успешного выполнения задачи зависят от анализируемого программного обеспечения, конкретного региона(ов), функциональности и целей исследования. Под успешным выполнением задачи (успехом) может пониматься как сложное действие, вроде завершения оформления товара в интернет-магазине, так и достаточно простое вроде отправки документа на печать.

Методики, ориентированные на результативность, потребуют добавления в онтологию таких классов как: задача, успех, ошибка и т.д. На данный момент, авторами представляется это излишним усложнением и требующим более тщательного экспериментального исследования. Поэтому, методики данной группы детально не описываются в рамках представленной работы.

Удовлетворенность оценивается как отношение к использованию продукта, так и восприятие пользователем таких показателей, как экономичность, полезность или легкость в изучении. Данная метрика во многом зависит от субъективной оценки. Поэтому, в настоящий момент она не рассматривается авторами в качестве возможной для формализации.

Далее рассмотрим детально методики, ориентированные на эффективность (efficiency-oriented), в виду их наибольшего потенциала, по мнению авторов.

5. Методики оценки удобства использования, ориентированные на эффективность

Эффективность – отношение израсходованных ресурсов к точности и полноте, с которой пользователи достигают поставленных целей.

Ориентированные на эффективность методики связывают успешность выполнения задач с затрачиваемыми ресурсами. Как и для результативности, существуют детализированные показатели эффективности. Стандарт ISO 9241-11:1998 [2] определяет, например, следующие показатели:

- время необходимое на завершение задачи;
- задачи, выполненные в единицу времени;
- финансовая стоимость выполнения задачи.

Вероятно, самой наглядной и известной методикой в данной категории является построение и анализ тепловых карт на основе выборки данных по действиям пользователей. Тепловые карты известны довольно давно и функционал для их построения представляется различным программным обеспечением и веб-сервисами. Однако, формат хранения данных активности пользователей обычно закрыт. Использование онтологии в данном случае упростит обмен данными активности пользователей за счет использования общей модели данных, представленной онтологией.

Ранее авторами был разработан метод построения тепловых карт на основе данных активности пользователей [12,13]. Кроме того, было разработано программное обеспечение для сбора и анализа данных активности пользователей на основе представленной онтологии [14]. Их совместная апробация проведена на базе программно-аппаратного комплекса «Лего-машина Тьюринга» [15], ранее разработанного авторами для демонстрации принципов работы абстрактного исполнителя. Программная часть указанного комплекса является прикладным программным обеспечением и относится к категории настольных приложений для операционных систем семейства Windows. В программную часть было внедрено программное обеспечения для сбора данных активности пользователей.

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.

Авторами в данный момент разрабатывается так же методика, нацеленная на поиск ранее неизвестных повторяющихся последовательностей команд. Наличие таких последовательностей может сигнализировать о существовании совокупностей команд, требующих реорганизации.

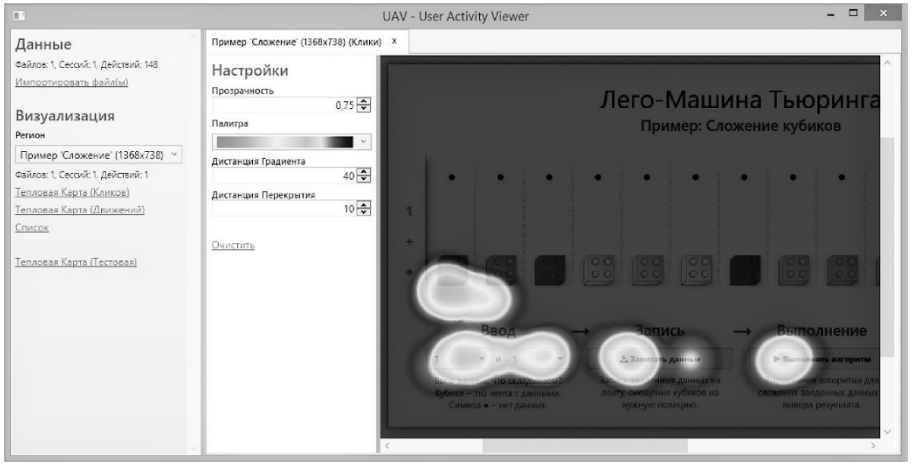

Рис. 6. Тепловая карта одного из регионов комплекса «Лего-машина Тьюринга»
Fig. 6. Heatmap of one of the regions of the “LEGO Turing machine” complex

Допустим, определено множество команд {1, 2, 3, 4}. Имеется выборка использованных команд в хронологическом порядке из данных активности пользователя за одну сессию:

{2, 1, 4, 1, 2, 3, 1, 2, 1, 1, 2}

Последовательность {1, 2} часто встречается в сессии, т.е. команды {1} и {2} часто выполняются последовательно. Данное наблюдение может сигнализировать о необходимости детального анализа указанных команд. Возможно, что команда {1} лишь открывает доступ к часто используемой {2} и необходимо повысить доступность {2}. Либо необходимо создать новую команду {1+2}, которая бы выполняла автоматически обе команды.

Разработанная онтология позволит проводить детальный анализ статистики затраченного времени и действий в конкретном регионе, возможно, с разбиением на различные категории пользователей или устройств. Обнаружение, например, избыточных действий (перемещений курсора), может сигнализировать о необходимости перегруппировки элементов пользовательского интерфейса. Возможен сбор и анализ различной статистики для ранее известных категорий пользователей, с целью лучшего их понимания. Например, выявление частоты использования «горячих клавиш».

Для выбора конкретного решения потребуется привлечение эксперта и экспертный анализ. Оно будет зависеть от типа программного обеспечения, его функциональности, категории пользователей и множества других факторов, которые должен будет учесть эксперт по удобства использования. Однако, применение подобных методик позволит автоматизированным образом выявить все участки программной системы, требующие детального анализа.

Это, в свою очередь, сэкономит время на поиски таких участков и во многом уменьшит субъективность при их отборе.

6. 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.

7. Заключение

Описанная версия онтологии «Удобство использования программного обеспечения» предназначена для представления данных активности пользователей прикладного программного обеспечения. Проведенные эксперименты показали, что онтология может быть использована для анализа удобства использования настольных приложений для операционных систем семейства Windows, и в том числе для построения тепловых карт.

Кроме того, онтология может быть использована и расширена любыми исследователями, занимающимися проблемами удобства использования программных интерфейсов, в частности в рамках экспериментов с различными типами интерфейсов. Авторы будут рады замечаниям и предложениям по

уточнению и развитию онтологии от специалистов в данной предметной области и готовы к сотрудничеству.

В данный момент проходит процесс публикации онтологии в вебе под открытой лицензией в одном из синтаксисов языка OWL, а также осуществляется разработка формальных методов ее использования для оценки удобства использования программных продуктов.

Список литературы

[1]. Федеральное агентство по техническому регулированию и метрологии, ГОСТ Р ИСО 9241-210–2016, "Эргономика взаимодействия человек-система – Часть 210: Человеко-ориентированное проектирование интерактивных систем". Информационный портал по стандартизации. Стандартинформ, 2017.

[2]. ISO 9241-11:1998, ISO 9241-11:1998. Ergonomic requirements for office work with visual display terminals (VDTs) – Part 11: Guidance on usability. ISO, URL: https://www.iso.org/obp/ui/#iso:std:iso:9241:-11:ed-1:v1:en (дата обращения 10.01.2017).

[3]. Федеральное агентство по техническому регулированию и метрологии ГОСТ-28806-90: Качество программных средств. Термины и определения. Информационный портал по стандартизации, Стандартинформ, 2017.

[4]. Данилов Н.А., Шульга Т.Э. Обзор моделей онтологий предметных областей «пользовательский интерфейс» и «юзабилити». Проблемы управления, обработки и передачи информации (УОПИ-2015): сб. тр. IV Междунар. науч. конф.: в 2 т. под ред. А.А. Львова и М.С. Светлова, Саратов: Издательский Дом «Райт-Экспо», 2015, том 1, стр. 214-218.

[5]. Linked Data Glossary. W3C Working Group Note 27 June 2013. URL: http://www.w3.org/TR/2013/NOTE-ld-glossary-20130627/#ontology (дата обращения 01.-02.2017).

[6]. Когаловский М.Р. Системы доступа к данным, основанные на онтологиях. Программирование, том 38, № 4, 2012 г., стр. 55-77.

[7]. Астраханцев Н.А., Турдаков Д.Ю. Методы автоматического построения и обогащения неформальных онтологий. Программирование, том 39, № 1, 2013 г., стр. 23-34.

[8]. Шульга Т.Э., Вагарина Н.С., Мельникова Н.И., Мищенко Д.А. Модели и инструменты представления пространственно-временных данных в семантическом вебе. Известия Самарского научного центра Российской академии наук, том 18, № 4-4, 2016 г., стр. 844-851.

[9]. G. Atemezing, J. Berners-Lee. A user interface ontology (ui). Linked Open Vocabularies, 2014, URL: http://lov.okfn.org/dataset/lov/vocabs/ui (дата обращения 20.10.2016).

[10]. Грибова В.В., Тарасов А.В. Модель онтологии предметной области «Графический Пользовательский Интерфейс». Интеллектуальные системы, 2005, №1(9).

[11]. A. Blandford, T. Green. OSM: an ontology-based approach to usability evaluation. Proceedings of Workshop on Representations. Queen Mary & Westfield College, 1997, Июль.

[12]. Danilov N., Shulga T., Frolova N., Melnikova N., Vagarina N., Pchelintseva E. Software usability evaluation based on the user pinpoint activity heat map. Advances in Intelligent Systems and Computing, vol. 465, 2016, pp. 217-225.

[13]. Данилов Н. А., Шульга Т. Э. Метод построения тепловой карты на основе точечных данных об активности пользователя приложения. Прикладная информатика, том. 10, № 2 (56), 2015 г., стр. 49-58.

[14]. Шульга Т.Э., Данилов Н.А. Свидетельство о государственной регистрации программ для ЭВМ № 2014662094 «Программный комплекс для сбора и визуализации данных активности пользователя настольного приложения» от 6 октября 2014 г.

[15]. Шульга Т. Э., Данилов Н. А., Валявский В. А., Митрофанов А. А., Мурзабеков А. М. Свидетельство о государственной регистрации программы для ЭВМ №2015610568 «Лего-машина Тьюринга» от 13 января 2015 г.

Ontology of the “Software Usability” Domain

A.A. Sytnik <as@sstu.ru>
T.E. Shulga <shulga@sstu.ru>
N.A. Danilov <Nikita_Danilov@outlook.com>
Yuri Gagarin State Technical University of Saratov, 77, Politechnicheskaya st., Saratov, 410054, Russia

Abstract. The article presents the ontology of the "Software usability" domain. Authors provides the review of existing ontologies for “user interface” and “software” domains. The article describes the advantages that can give its use in the analysis and evaluation of software products usability, e.g.: opened data structure for more easily data sharing. The paper presents a class diagram of the proposed ontology and a text description for the classes, object properties and data properties. Presented diagrams contains basic classes (device, session, user, region, variation, image) and subclasses for the event class. Examples of questions that can be answered by ontology are given. e.g.: “What commands are made by users in the region?” and “What variations of the region are in the session?”. Also, the article presents the SPARQL queries that may be used for this ontology. The classification of possible methods of analysis and evaluation of usability on the basis of the ontology is proposed and some of them are described. An example of the user activity data in a standard RDF/XML format is demonstrated. Described ontology "Software Usability" designed for the user activity data representation. Experiments have shown that ontology can be used to analyze the usability of desktop applications for Windows operating systems, including the construction of heat maps. Heat map based on the collected user activity data in is shown. In addition, ontology can be used and extended by any researchers involved in the problems of ease of use of software interfaces, in particular in experiments with different types of interfaces. The authors will welcome comments and suggestions on the refinement and development of ontology from experts in the domain and are ready to cooperate.

Keywords: software usability; user interface; ontology; ontological engineering.

DOI: 10.15514/ISPRAS-2018-30(2)-10

For citation: Sytnik A.A., Shulga T.E., Danilov N.A. Ontology of the “Software Usability” Domain. Trudy ISP RAN/Proc. ISP RAS, vol. 30, issue. 2, 2018, pp. 195-214 (in Russian). DOI: 10.15514/ISPRAS-2018-30(2)-10

References

[1]. Federal Agency for Technical Regulation and Metrology, GOST R ISO 9241-210–2016, "Ergonomics of human-system interaction. Part 210. Human-centred design for interactive systems". Standardization information portal. Standartinform, 2017 (in Russian).

[2]. ISO 9241-11:1998, ISO 9241-11:1998. Ergonomic requirements for office work with visual display terminals (VDTs) – Part 11: Guidance on usability. ISO, URL: https://www.iso.org/obp/ui/#iso:std:iso:9241:-11:ed-1:v1:en.

[3]. Federal Agency for Technical Regulation and Metrology, GOST-28806-90: Software quality, Terms and definitions. Standardization information portal, Standartinform, 2017 (in Russian).

[4]. Danilov N.A., Shulga T.E. Review of ontology models of "user interface" and "usability" domains. Problemy upravlenija, obrabotki i peredachi informacii (UOPI-2015): sb. tr. IV Mezhdunar. nauch. konf.: v 2 t. pod red. A.A. L'vova i M.S. Svetlova [Proc. of conference “Problems of information management, processing and transmission (UOPI-2015)”], Saratov: Publishing House «Rajt-Jekspo», 2015, vol. 1, pp. 214-218 (in Russian).

[5]. Linked Data Glossary. W3C Working Group Note 27 June 2013. URL: http://www.w3.org/TR/2013/NOTE-ld-glossary-20130627/#.

[6]. Kogalovsky M.R. Ontology-based data access systems. Programming and Computer Software, vol. 38, issue 4, 2012, pp. 167-182. DOI: 10.1134/S0361768812040032.

[7]. Astrakhantsev N.A., Turdakov D.Y. Automatic construction and enrichment of informal ontologies: A survey. Programming and Computer Software, vol. 39, issue 1, 2013, pp. 34-42. DOI: 10.1134/S0361768813010039.

[8]. Shulga T.E., Vagarina N.S., Melnikova N.I., Mishhenko D.A. [Models and tools for presenting spatiotemporal data in a semantic web], Izvestija Samarskogo nauchnogo centra Rossijskoj akademii nauk, 2016, vol. 18, issue 4-4, pp. 844-851.

[9]. G. Atemezing, J. Berners-Lee. A user interface ontology (ui). Linked Open Vocabularies, 2014, URL: http://lov.okfn.org/dataset/lov/vocabs/ui (acessed 20.10.2016).

[10]. Gribova V.V., Tarasov A.V. The ontology model of the “Graphical User Interface” domain. [Intelligent system], Intellektual'nye Sistemy, 2005, issue 1(9) (in Russian).

[11]. A. Blandford, T. Green. OSM: an ontology-based approach to usability evaluation. Proceedings of Workshop on Representations. Queen Mary & Westfield College, 1997, Июль.

[12]. Danilov N., Shulga T., Frolova N., Melnikova N., Vagarina N., Pchelintseva E. Software usability evaluation based on the user pinpoint activity heat map. Advances in Intelligent Systems and Computing, 2016, vol. 465, pp. 217-225 (in Russian).

[13]. Danilov N.A., Shulga T.E. Constructing a heat map based on the point data of the application user’s activity. Applied informatics, 2015, vol. 10, issue 2 (56), pp. 49-58 (in Russian).

[14]. Shulga T.E., Danilov N.A. Certificate of state registration of computer programs № 2014662094 "Software package for the collection and visualization of the desktop application user activity data», 2014, 6 October.

[15]. Shulga T. Je., Danilov N. A., Valjavskij V. A., Mitrofanov A. A., Murzabekov A. M. Certificate of state registration of computer programs №2015610568 «LEGO Turing machine», 2015, 13 January.