УДК 004.4'2
DOI:

РАЗРАБОТКА ИНСТРУМЕНТОВ ВИЗУАЛИЗАЦИИ ДАННЫХ НА ОСНОВЕ ПРЕДМЕТНО-ОРИЕНТИРОВАННОГО МОДЕЛИРОВАНИЯ
Development of Data Visualization Tools Based on Domain Specific Modeling

А. Д. Джейранян[1], И. Д. Ермаков[2], К. А. Проскуряков[1], Л. Н. Лядова[1]
[1]Национальный исследовательский университет «Высшая школа экономики», Пермь, Россия
[2]Пермский государственный национальный исследовательский университет, Пермь, Россия

A. D. Dzheiranian[1], I. D. Ermakov[2], K. A. Proskuryakov[1], L. N. Lyadova[1]
[1]National Research University – Higher School of Economics (HSE University), Perm, Russia
[2]Perm State National Research University (PSU), Perm, Russia

Аннотация. Описывается подход к разработке средств визуализации данных, обеспечивающий возможность настройки на потребности пользователей и специфику предметных областей, в которых они работают, основанный на предметно-ориентированном моделировании. Кратко представлены результаты анализа инструментов визуализации данных и возможности их настройки на предметные области исходя из потребностей пользователей и решаемых ими задач. Показано, что существующие инструменты требуют от пользователей навыков программирования для настройки формата визуализации данных или разработки новых моделей визуализации. Предлагается использовать инструменты предметно-ориентированного моделирования (языковой инструментарий) для создания предметно-ориентированных языков (DSL), предназначенных для разработки новых моделей визуализации данных, отражающих специфику решаемых пользователями задач. Использование разработанных инструментов не требует от пользователей профессиональных знаний языков программирования. Описывается архитектура программной системы, управляемой знаниями. Ядро системы – многоаспектная онтология, которая включает описания языков и предметных областей, а также правила генерации новых языков и трансформации построенных моделей. Языки предназначены для описания различных классов диаграмм. Система включает инструменты для автоматизации создания новых DSL через отображение онтологии предметной области на метамодель базового языка по заданным пользователем правилам, которые также сохраняются в онтологии. Классификация различных типов диаграмм составляет основу для создания онтологии языков визуализации данных. Описывается пример базового языка для создания диаграмм. Демонстрируется возможность настройки DSL и трансформации разработанных с его помощью моделей визуализации, генерации кода, реализующего модель.

***Ключевые слова*:** визуализация данных, предметно-ориентированное моделирование, предметно-ориентированный язык, метамоделирование, грамматика, многоаспектная онтология, трансформация моделей

Аннотация. An approach to the development of data visualization tools is described that provides the ability to customize to the needs of users and the specifics of the domains in which they work, based on domain-specific modeling. The results of the analysis of data visualization tools and the possibility of customizing them to subject area based on the needs of users and the tasks they solve are briefly presented. It is shown that existing tools require programming skills from users to customize the data visualization format or to develop new visualization models. It is proposed to use domain-specific modeling tools (language toolkits) to create domain-specific languages (DSL) designed to develop new data visualization models that reflect the specifics of the tasks solved by users. The use of the developed tools does not require users to have professional knowledge of programming languages. The architecture of a knowledge-driven software system is described. The core of the system is a multifaceted ontology, which includes descriptions of languages and domains, as well as rules for generating new languages and transforming constructed models. Languages are designed to describe different classes of diagrams. The system includes tools for automating the creation of new DSLs by mapping the domain ontology onto the metamodel of the base language according to user-specified rules, which are also stored in the ontology. The classification of different types of diagrams forms the basis for creating an ontology of data visualization languages. An example of a basic language for creating diagrams is described. The ability to customize the DSL and transform visualization models developed with its help and generate code that implements the model is demonstrated.

***Ключевые слова*:** data visualization, domain-specific modeling, domain-specific language, metamodeling, grammar, multifaceted ontology, model transformation.

I. Введение

Инструменты визуализации получили широкое распространение в различных отраслях, бизнес-функциях и ИТ-дисциплинах, как в частном, так и в государственном секторе. Они активно используются в таких областях, как энергетика, картография, медицина, финансы, социология и многие другие. В этом контексте визуализация данных служит методом анализа данных. Эффективность работы экспертов, аналитиков во многом определяется качеством визуализации данных, результатов их анализа.

Вопросам качества визуализации данных посвящены многочисленные исследования.

Нечитабельные, запутанные визуализации распространяют дезинформацию, вводят в заблуждение и снижают эффективность работы исследователей, качество управленческих решений и т. д. Авторы статьи [1] подчёркивают масштаб проблемы, ссылаясь на ресурсы, собирающие десятки тысяч пользователей, критикующих графики и диаграммы (reddit.com/r/dataisugly), выполненные с очевидными ошибками. Существующие методы визуализации данных дают возможность строить в основном простые и чрезмерно обобщенные визуализации, которым, к тому же, часто не хватает систематичности и универсальности. Отсутствие надлежащего инструментария для оценки качества визуализации данных еще больше усугубляет проблему. В статье предлагается VisQualdex – систематический набор рекомендаций по статической визуализации данных. Категоризация основана на теории грамматик графики. Предлагаются десятки критериев, предназначенных для выявления различных ошибок (ошибок разных категорий и масштаба). Предложенный набор рекомендаций рецензирован и протестирован экспертами в областях визуализации данных, науки о данных, графического дизайна, информационных технологий и информатики. Реализация рекомендаций доступна в виде веб-сервера, разработанного как одностраничное приложение на JavaScript с использованием принципов Vue.js и Material Design.

Особое внимание исследователями уделяется вопросам применения средств визуализации в образовании.

Авторы статьи [2] показывают, что инфографика важна для представления и передачи сложной информации, в частности, обсуждают влияние визуализации на результаты учебного процесса, организованного при взаимодействии с инфографикой, мультимедийными средствами. Результаты исследования показали, что использование инфографики, визуального обучения является эффективным подходом к организации учебного процесса. Цель исследования, результаты которого представлены в [3], – изучение теоретических и практических аспектов визуализации знаний в образовательном процессе. Авторы сравнивают различные модели представления знаний, рассматривают принципы визуализации и структуру компетенций визуализации, включающую аналитические, алгоритмические и пр. компоненты. Используя различные методы моделирования, визуализации и систематизации данных, авторы делают вывод, что визуализация знаний требует учёта особенностей образовательного процесса в контексте дидактических принципов и пр. Формирование навыков визуализации в ходе учебной деятельности – это сложная задача, зависящая от множества факторов. Эти навыки являются базовыми компонентами формирования визуализационной компетентности. Решение задачи визуализации знаний требует использования современных инновационных технологий, охватывающих базовые цифровые технологии, технологии обработки больших данных, методы интеллектуального анализа многомерных данных и т. д. В статье [4] рассматриваются программные средства, которые используются в образовательном процессе и влияют на формирование профессиональных компетенций у студентов. Цель проекта – разработка технологии создания образовательного VR-контента, который должен повысить эффективность обучения с помощью множества различных программных сред, используемых в процессе формирования конечного виртуального пространства.

В статье [5] описывается подход к пониманию идеи когнитивной ясности графовых моделей. Авторами описывается концептуальная схема структурирования понятий, связанных с когнитивной ясностью. В соответствии с этой схемой они выделяют факторы формирования когнитивной ясности. Когнитивная ясность определяется как набор внутренних характеристик визуального образа модели, эффекты присутствия которых проявляются при визуальном анализе модели. Авторы делают вывод, что наибольший интерес представляют именно факторы формирования когнитивной ясности (в силу их конструктивности). В статье приводится детализированная схема подхода к пониманию идеи когнитивной ясности и обсуждаются отдельные компоненты этой схемы. Предлагается обобщенный алгоритм подготовки и проведения эксперимента по решению аналитиком определенной задачи визуального анализа с участием факторов, о влиянии которых перед проведением эксперимента формируется гипотеза. В результате оценивается эффект изменения уровня когнитивной ясности и выявляется характер зависимости от заданных факторов или ее отсутствие, что позволяет принять, отвергнуть или уточнить исходную гипотезу.

Вопросы эффективности визуализации данных рассматриваются в статьях [6, 7]. Восприятие графической информации связано с определёнными зрительными когнитивными процессами, опираясь на исследование которых, авторы дают общие рекомендации по созданию эффективных научных визуализаций.

Потребности конечных пользователей (аналитиков данных) включают необходимость создания пользовательских типов диаграмм для конкретных задач и предметных областей, поскольку базовые типы диаграмм с базовой геометрией могут ограничивать передачу информации [1] и приводить к неэффективной визуализации, что, в свою очередь, может привести к ошибкам в принятии решений [8].

Пользователям требуется создание визуализаций, адаптированных к специфике решаемых ими задач и предметных областей. Таким образом, существует необходимость внедрения пользовательских спецификаций визуализации в используемые средства. Эти спецификации определяют, как пользователи могут указывать свои требования для создания визуализаций [9]. Статическая визуализация может оказаться неэффективной для решения задач анализа больших наборов данных, поэтому существует потребность в быстром и простом создании интерактивных визуализаций [10]. Эти методы должны быть способны работать с различными типами и источниками данных [9].

Исследователи [8, 11] отмечают ограниченную степень гибкости в манипулировании элементами диаграмм и отсутствие ориентированности на реальные потребности пользователя в существующих инструментах визуализации данных. Часто такая настройка требует использования языка программирования [10]. Отсутствие глубоких знаний программирования у пользователей приводит к необходимости создания no-code или low-code платформ.

В статье [12] рассматриваются различные техники и инструменты визуализации данных, приводится их классификация, формулируются проблемы, с которыми сталкиваются пользователи, и новые возможности, которые должны быть обеспечены перспективными средствами визуализации. В частности отмечается, что одной из ключевых задач является разработка методов автоматизации визуализации данных, обеспечения эффективной и интерактивной визуализации независимо от размера и сложности данных. Реализация этих методов облегчает исследования в областях с интенсивным использованием данных [13], позволяет непрерывно отслеживать и анализировать эти данные, визуализировать результаты анализа.

Автоматический визуальный анализ предполагает решение задач поиска, очистки, интеграции и визуализации данных. Для решения этих задач широко используются онтологии.

В работах [13, 14] было предложено использовать онтологии как часть архитектуры, как ядро аналитической платформы, управляемой знаниями. В данном случае используется многоаспектная онтология, которая позволяет реализовать семантический поиск и индексацию данных, избежать их дублирования, расширять функциональность информационных и аналитических систем, создавать предметно-ориентированные языки и модели и сценарии исследований, а также автоматически интерпретировать данные и результаты анализа данных для предоставления их разным группам пользователей согласно знакомой им терминологии.

Доступные системы инструментов визуализации данных можно разделить на следующие группы:

1) электронные таблицы (например, Excel, Google Sheets),

2) аналитические платформы (например, Microsoft Power BI, Tableau),

3) редакторы диаграмм (например, Miro, ChartBlocks).

Стандартные инструменты первых двух групп ограничены базовыми типами диаграмм и возможностями настройки визуальных эффектов. Инструменты третьей группы позволяют создавать собственные визуализации, редактировать расположение элементов, но не предоставляют возможность настройки на предметные области.

Использование языков программирования общего назначения (например, библиотек Python для визуализации данных: Matplotlib, Seaborn, Plotly и др.) способствует созданию выразительных визуализаций для решения конкретных задач, но требует глубоких знаний программирования от разработчиков диаграмм. Также созданное решение нельзя повторно использовать для других визуализаций, и оно представляет собой «черный ящик», где непонятно, как настроена визуализация [10].

Визуализация, отражающая специфику предметной области, может в значительной степени способствовать лучшему пониманию представленных данных, результатов их анализа, формальных моделей. Создание таких средств может быть основано на предметно-ориентированном моделировании (Domain Specific Modeling, DSM), предполагающем создание и использование предметно-ориентированных языков (Domain Specific Language, DSL) для разработки новых вариантов (моделей) визуализации данных.

Публикаций по созданию DSL для визуализации данных немного. Судя по обзору, большинство DSL фокусируются на небольшом наборе стандартных диаграмм (круговых диаграмм, гистограмм и т. д.) или визуализации конкретных типов данных (например, геопространственных и т. п.). Они различаются уровнями абстракции, контекстами использования и возможностями реализации.

В статье [15] описан процесс разработки DSL для построения и преобразования методов визуализации данных. Этот DSL встроен в язык программирования Haskell. Авторы предоставляют несколько уровней абстракции: на самом низком уровне пользователь может создать элемент, состоящий из определенной примитивной формы и набора визуальных параметров. Важно отметить, что основные конструкции языка

ограничены гистограммой и круговой диаграммой. Однако допускается располагать их элементы по-разному для создания более сложных вариантов визуализации.

В статье [16] представлена вариационная модель визуализации, реализованная с помощью DSL, встроенного в язык программирования PureScript. Этот DSL позволяет создавать вариационные визуализации и их комбинации, например наложение альтернативных гистограмм. В статье также обсуждаются методы представления вариаций и добавления вариаций к визуализациям через DSL. Разработанные средства обеспечивают создание, управление, навигацию и рендеринг различных визуализаций.

Исследователи в статье [17] представляют DSL, ориентированный на геовизуализацию данных. Они используют компилятор для облегчения автоматического создания визуализаций и предварительной обработки данных. Их система использует возможности многоядерного параллелизма для ускорения предварительной обработки данных.

В статье [18] предлагается подход, который помогает пользователю создавать визуализации моделей для конкретной предметной области с использованием CSP (Communicating Sequential Processes) – формального языка, который в основном используется для описания параллельных и распределенных систем. Авторами успешно созданы различные визуализации моделей CSP, демонстрирующие возможности предлагаемого подхода. Однако рассматриваемые средства не являются универсальными, возможности языка ограничивают его использование.

Языково-ориентированный подход (в данном случае подход, основанный на DSM) к реализации средств визуализации данных может стать основным при разработке системы визуализации данных с возможностью настройки на потребности пользователей. Создание новых вариантов визуализаций, настроенных на конкретные предметные области и задачи пользователей, разбивается на два этапа:

1. Разработка предметно-ориентированного языка (DSL), отражающего потребности пользователей.
2. Разработка предметно-ориентированных моделей визуализации с использованием созданного языка.

Рассмотренные инструменты предоставляют возможность разработки новых типов диаграмм. Но возможности настройки на различные предметные области в них отсутствуют или они ограничены. Кроме того, от пользователей требуются профессиональные навыки программирования, что также ограничивает возможности использования описанных средств.

Для создания DSL используются DSM-платформы (языковые инструментарии), которые автоматизируют разработку редакторов моделей и средств их трансформации, генераторов кода. Актуальной становится задача снижения трудоёмкости создания самих языков, автоматизации их разработки.

Таким образом, конечные пользователи сталкиваются с двумя вопросами при решении задач визуализации данных:

1. Как обеспечить создание или настройку визуализаций в соответствии с потребностями пользователей?
2. Как снизить уровень требований к компетенциям пользователя в программировании при использовании средств визуализации, обеспечивающих возможность настройки на потребности пользователей?

Для расширения возможностей инструментов визуализации предлагается языково-ориентированный подход, использование языкового инструментария, управляемого знаниями, позволяющего автоматизировать создание предметно-ориентированных языков на основе использования многоаспектной онтологии [19, 20, 21].

II. Постановка задачи

Цель проекта, результаты которого представлены в данной статье, – разработать подход к созданию средств эффективной визуализации данных с учётом рекомендаций, предложенных в [6, 7], обеспечивающих возможность создания пользовательских типов диаграмм для конкретных задач и предметных областей, а также возможность создания интерактивных визуализаций [10].

Специфика требований заключается в необходимости настройки визуализации на специфику предметных областей и решаемых пользователями задач или разработки новых типов диаграмм. Этого результата можно достичь за счет интерактивности, комбинирования разных типов диаграмм и т. д. Эти требования должны быть учтены при создании средств для разработки моделей визуализации (графиков, схем, диаграмм).

Предлагаемая в статье основа для реализации инструментов визуализации данных – DSM-платформа, управляемая знаниями, обеспечивающая возможность создания иерархий предметно-ориентированных языков [19, 20, 21], предназначенных для создания новых типов моделей (новых графических нотаций), отвечающих потребностям пользователей. Снижение трудоёмкости разработки новых DSL достигается за счёт использования базовых языков, их дополнения и комбинирования в соответствии с правилами, определяемыми пользователями. Построенные с использованием созданных DSL модели визуализации могут быть переведены в программный код, реализующий эффективную пользовательскую визуализацию данных, на основе заданных правил трансформации типа «Модель-Текст».

Основные функциональные требования к инструментам визуализации данных показаны в формате диаграммы прецедентов на рис. 1. Эксперт в предметной области должен разработать онтологию предметной области, в которой пользователи – разработчики визуализаций – решают свои задачи. DSM-эксперт отвечает за разработку метамоделей базовых языков (в данном случае – языков для разработки базовых типов диаграмм, которые могут послужить основой для создания новых DSL, реализующих потребности пользователей), а также за разработку правил генерации новых языков, отражающих специфику предметных областей, в которых работают пользователи, – правил отображения онтологии предметной области на метамодель выбранного базового языка. При разработке визуализации пользователь может использовать созданный для неё DSL, выполнив настройку параметров, или разработать новый DSL. Генерация кода на языке программирования по построенной модели визуализации позволяет создавать интерактивную визуализацию, управляемую событиями, комбинировать различные типы визуализаций при работе со сложными структурами данных, получаемых из различных источников. Правила трансформации (генерации кода) разрабатываются DSM-экспертом.

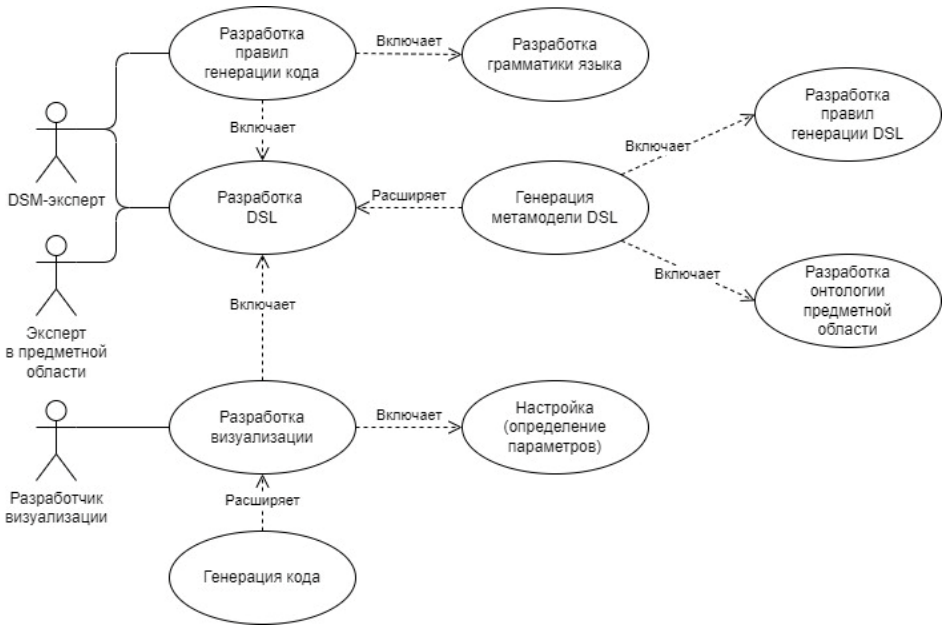


Рис. 1. Основные функции средств разработки моделей визуализации на основе DSM-подхода

Ядром языкового инструментария, с помощью которого должны быть реализованы описанные функции, является многоаспектная онтология [19, 20, 21], которая кроме описания предметной области включает и онтологию языков, где визуальные языки (языки создания схем, диаграмм и т. п.) описываются с помощью метамоделей, а текстовые – их грамматиками. Правила генерации DSL и кода также включаются в онтологию.

Разработка метамоделей DSL, правил генерации новых DSL и генерации кода не требует знания языков программирования – пользователи работают с редакторами и конструкторами в визуальной среде. Для создания онтологии предметной области предлагается использовать метод Mask [22]. Таким образом снижаются требования к квалификации пользователей, к навыкам программирования.

Для реализации предлагаемого подхода необходимо решить следующие задачи:

1. Проанализировать различные типы диаграмм и построить онтологию языков визуализации данных на основе классификации диаграмм.
2. Разработать метамодели базовых языков для различных типов диаграмм, используемых для визуализации данных, – основу для создания новых типов визуализации.
3. Разработать правила генерации новых DSL – новых языков визуализации данных, отражающих специфику предметных областей, и сгенерировать новые типы диаграмм на основе разработанных правил.
4. Разработать правила трансформации (генерации кода) для реализации нового типа визуализации данных.

Результаты необходимо проиллюстрировать примерами визуализации данных с использованием разработанных средств.

III. Теория

Визуальные языки основаны на абстрактных – графовых – моделях: любая модель, диаграмма представляется как граф, в котором представлены объекты некоторой предметной области и связи между ними; абстрактная графовая модель расширяется – дополняется атрибутами, характеристиками, которые могут быть приписаны

элементам графа, представляющим объекты конкретной предметной области; для этой конкретизированной модели строится её визуальное представление с использованием конкретного типа визуализации – графической нотации, которую пользователь выбирает, основываясь на своих потребностях. Таким образом, модель с одной и той же структурой может по-разному визуализироваться при решении пользователями различных задач.

1. *Графовая модель для реализации методов визуализации данных на основе DSM-подхода*

В основе реализации языкового инструментария – графовая модель, которая должна, с одной стороны, обладать достаточными выразительными возможности для представления структуры моделируемых объектов в различных предметных областях, а с другой – позволять разрабатывать эффективные алгоритмы для операций над построенными моделями.

Для реализации языкового инструментария предложена графовая модель – иерархические гиперграфы с полюсами (*HP*-графы) [19].

HP-граф – это ориентированный граф, который представляется как тройка $G = (P, V, W)$, где $P = \{\pi_1, \ldots, \pi_n\}$ – это множество внешних полюсов графа, $V = \{v_1, \ldots, v_m\}$ – это непустое множество вершин, а $W = \{w_1, \ldots, w_l\}$ – это множество гиперрёбер графа. Пусть *Pol* – абстрактное множество всех полюсов графа. Полюс – это базовое понятие выбранной графовой модели, через которое определяются все элементы графа:

- *Вершины* $v \in V$ – это непересекающиеся подмножества *полюсов* из множества *Pol*, при этом объединение всех этих подмножеств даёт всё множество полюсов *Pol*, то есть каждый полюс принадлежит ровно одной вершине. Полюсы – это «интерфейс» вершины, через который вершина связывается с другими вершинами. Каждая вершина $v \in V$ представленная множеством полюсов $P_v = \{p_{v_1}, \ldots, p_{v_n}\}$, имеет множество входных и выходных полюсов ($I(v)$ и $O(v)$), эти множества могут пересекаться, то есть один и тот же полюс может быть и входным, и выходным. В простейшем случае вершина может быть представлена единственным полюсом, тогда $I(v) = O(v)$.
- Множество *внешних полюсов* P графа – это также подмножество множества полюсов *Pol*. Это множество состоит из входных и выходных полюсов графа (эти полюсы являются «входами» и «выходами» графа): $P = I(G) \cup O(G)$, эти полюсы являются входными или выходными полюсами вершин.
- Каждое *гиперребро* графа $w \in W$ определяет связи между вершинами и представляется подмножеством множества полюсов этих вершин: $w = P_w = \{p_{w_1}, \ldots, p_{w_k}\}$, каждое такое множество должно содержать хотя бы один входной и один выходной полюс (эти полюсы могут принадлежать нескольким вершинам, между которыми устанавливаются связи, или одной вершине).

Как вершины, так и рёбра могут быть «раскрыты» – расшифрованы графом, который детализирует их структуру. Таким образом может быть построена иерархическая диаграмма (с декомпозицией её элементов, которая может быть выполнена в интерактивном режиме, по правилам, заданным пользователем).

Показано, что эта модель обладает всеми возможностями, которые обеспечивают другие широко используемые графовые модели (метаграфы, гиперграфы, графы с полюсами и пр.), а также позволяет оптимизировать алгоритмы – минимизировать их временную сложность, упростить объектную модель при реализации программных средств. Иерархическая модель строится с помощью операции расшифровки, определённой для этого типа графов (операции декомпозиции, которая может быть применена как к вершинам, так и к гиперрёбрам). Такие возможности графовой модели обеспечивают эффективную реализацию группировки элементов при построении диаграмм, декомпозиции при разработке интерактивных визуализаций.

2. *Метаязык для описания визуальных языков в языковом инструментарии*

При разработке визуальных языков в языковых инструментариях строятся их описания в виде графовых грамматик или метамоделей. Для создания этих описаний должны использоваться соответствующие метаязыки (по аналогии с формальными грамматиками, которые можно задать в форме БНФ (форма Бэкуса–Наура) или диаграмм Вирта). Метаязык должен обеспечивать возможность представления *HP*-графа, всех его элементов. Оценка метаязыков представлена в табл. 1. Наиболее подходящим метаязыком с точки зрения представления всех элементов, которые могут быть использованы в графической модели, является язык *GOPPRR* (Graph, Object, Port, Property, Relationship, Role) DSM-платформы MetaEdit+, где Graph представляет модель в целом, Object – элемент модели, представляемый вершиной, Port представляет точку, через которую вершина-объект связывается с другими вершинами, Property – это свойство вершины или связи, Relationship – связь между объектами, Role – роль вершины в связи.

Ни один из языков не позволяет описывать в модели «обобщённые» связи, в которых могут участвовать множества вершин.

Для реализации языкового инструментария предложен язык [21], который является расширением GOPPRR.

ТАБЛИЦА 1
СРАВНЕНИЕ МЕТАЯЗЫКОВ, ИСПОЛЬЗУЕМЫХ В ЯЗЫКОВЫХ ИНСТРУМЕНТАРИЯХ

Метаязык	**Модель**	**Объект модели**	**Связи между объектами**	**Роли**	**Полюсы**	**Свойства**
EMOF	–	Class	Property, Association	–	–	Property
Ecore	–	EClass	EReference	–	–	EAttribute
GOPPRR	Graph	Object	Relationship	Role	Port	Property
ArkM3	–	Class	Association, Composition	–	–	Property
WebGME MML	FCO	FCO	Pointer, Set, Connection, Containment, Inheritance, Mixin	Connection Role	–	Attribute

Разработанный метаязык (рис. 2) позволяет создавать с его помощью метамодели визуальных языков с использованием всех описанных выше элементов *HP*-графов.

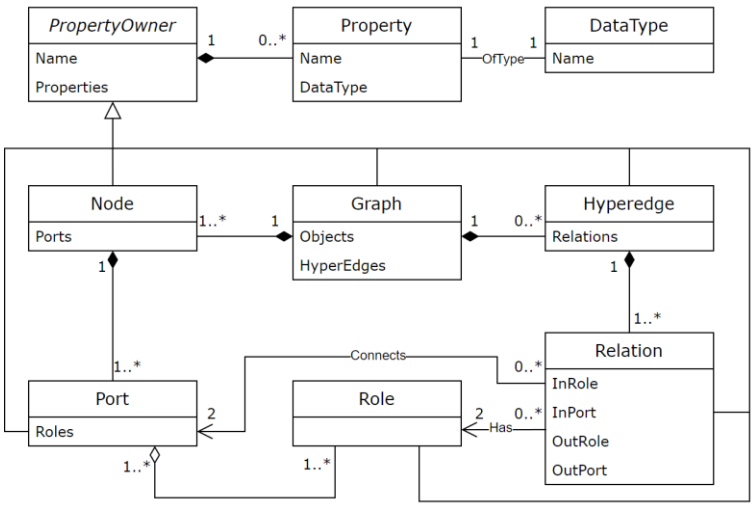


Рис. 2. Описание метаязыка *HPGPR* исследовательского прототипа языкового инструментария

Описанные выше модели использованы при разработке исследовательского прототипа языкового инструментария, позволяющего создавать новые предметно-ориентированные языки (визуальные DSL), настраиваемые на потребности пользователей, на основе базовых языков, метамодели которых представлены в системе. Разработанные языки также могут использоваться в качестве базовых для создания новых языков.

3. Обобщённая структура языкового инструментария

Обобщённая структура языкового инструментария, использованного для разработки прототипа средств визуализации данных, основанных на языково-ориентированном подходе, представлена на рис. 3. Показаны только те компоненты, которые должны быть реализованы для создания настраиваемых средств визуализации данных на основе DSM-подхода. В состав *многоаспектной онтологии* входят:

– *онтология визуальных языков* моделирования, включающая метамодели базовых языков, предназначенных для разработки визуальных моделей, описанные при помощи метаязыка платформы *HPGPR*, а также языков (DSL), созданных на их основе;

– *онтология текстовых языков*, содержащая описание грамматик текстовых языков;

– *онтология правил трансформаций*, хранящая двойки, первыми элементами которых являются элементы метамоделей визуальных языков, а вторыми – нетерминальные символы конструкций текстовых языков;

– *онтология предметных областей*, содержащая информацию о предметных областях, в которых пользователи решают свои задачи, для которых создаются DSL;

– *онтология правил генерации метамоделей* новых DSL, отражающих специфику предметных областей, включающая описание правил проецирования онтологий предметных областей на метамодели визуальных языков, выбранных в качестве базовых;

– *онтология моделей*, содержащая информацию о созданных моделях;

– *онтология пользовательских действий*, хранящая информацию о пользовательских действиях при работе с моделями (например, клики или наведения курсора мышки);

– *онтология взаимодействия*, представляющая информацию о возможных действиях пользователей при

работе в интерактивном режиме в системе, например, построения новой модели, или приближения её части и пр.;

– *онтология правил интерпретации*, хранящая тройки, первыми элементами которых являются элементы метамоделей визуальных языков, вторыми элементами – пользовательские действия, а третьими – интерактивные действия, которые могут быть использованы при разработке интерактивных визуализаций.

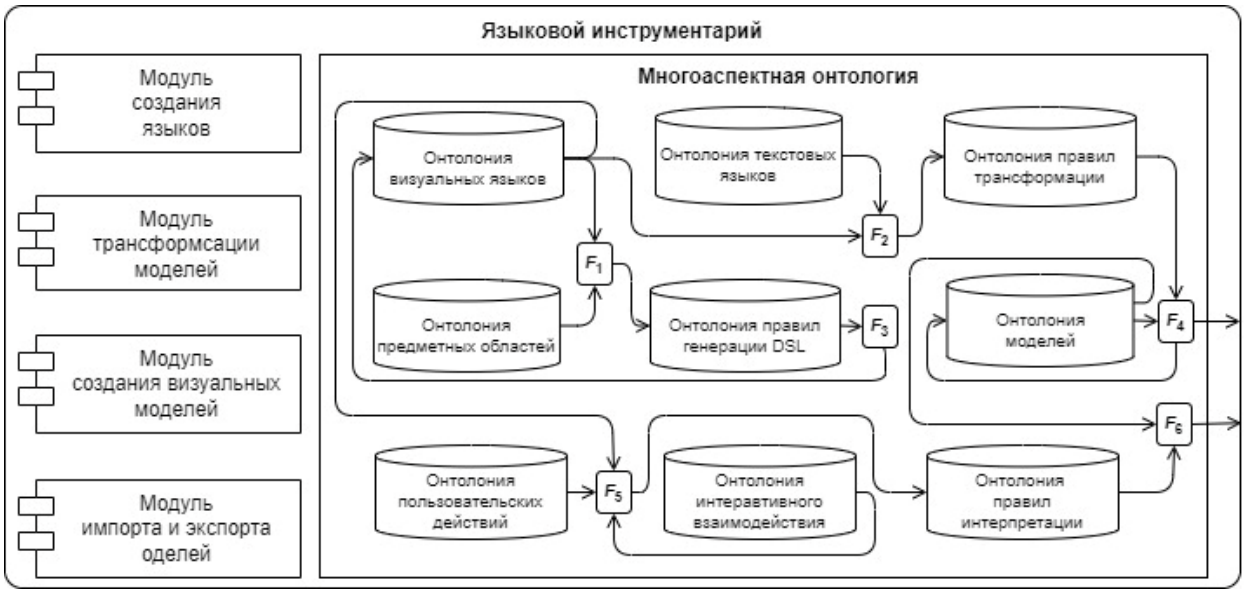


Рис. 3. Упрощённая структура языкового инструментария, управляемого знаниями

Многоаспектная онтология включает также функции, реализующие базовые алгоритмы, обеспечивающие автоматизацию генерации метамоделей языков, трансформацию моделей (в частности, генерацию кода – трансформацию моделей типа «Модель–Текст»), создание интерактивных моделей, их интерпретацию:

– F_1 – функция, обеспечивающая автоматизацию разработки новых языков путём отображения элементов онтологии предметной области на элементы метамодели выбранного базового языка, создания правил генерации DSL. Результат – правила генерации (отображения), определённые пользователем с помощью специального конструктора.

– F_2 – функция, с помощью которой определяется соответствие конструкций текстовых языков элементам метамоделей визуальных языков для создания правил трансформации моделей (генерации кода).

– F_3 – функция, реализующая генерацию метамодели нового DSL в соответствии с правилами, полученными с помощью F_1. Метамодель нового DSL сохраняется в онтологии визуальных языков. Новый DSL – подкласс базового языка, к которому применены правила отображения онтологии предметной области на его метамодель, заданные пользователем при выполнении F_1.

– F_4 – функция, реализующая трансформации моделей в соответствии с заданными правилами (для текстовых целевых языков выполняется генерация кода – трансформация типа «Модель-Текст»).

– F_5 – функция, принимающая элементы метамоделей визуальных языков, пользовательские сценарии и интерактивные действия для создания правил интерпретации для интерактивных моделей (визуализаций).

– F_6 – функция, которая обеспечивает применение заданных правил интерпретации к визуальным моделям для предоставления пользователям возможностей взаимодействия с моделями, создания интерактивных визуализаций.

Многоаспектная онтология включает также онтологию источников данных, онтологию функциональных модулей, которые также используются при создании моделей визуализации.

Далее описаны результаты разработки пользовательского прототипа средств визуализации данных, настраиваемых на потребности пользователей, основанные на использовании описанного выше подхода.

IV. Результаты экспериментов

Для апробации подхода (реализации прототипа средств визуализации) необходимо построить онтологию языков и метамодели базовых языков для описания диаграмм (визуализации данных традиционными способами), задать правила генерации новых DSL, с помощью которых можно выполнить визуализацию данных, отражающую потребности пользователей, определить правила трансформации моделей для создания интерактивной визуализации. Далее приведено краткое описание результатов решения этих задач.

1. *Классификация методов визуализации*

Разнообразие способов визуализации достаточно велико и продолжает расширяться, что подтверждается постоянным появлением новых специализированных типов диаграмм (древовидных, хордовых, сетевых и т. д.).

Выбор наиболее подходящего типа визуализации в первую очередь определяется тем, какую задачу решает пользователь, какую идею он должен передать с помощью диаграммы [23, 24]. Чтобы определить, какие именно методы визуализации достаточны для реализации большинства идей визуализации, необходимо остановиться на определенной таксономии – выполнить классификацию методов визуализации данных.

Было решено сосредоточиться на пятикатегорийной структуре, предложенной Энди Кирком [24]:

1) сравнение категорий;
2) оценка иерархий и отношений части к целому;
3) представление изменений с течением времени;
4) определение типов связей и отношений;
5) картографирование геопространственных данных.

При разработке классификации учтены следующие характеристики:

1) степень популярности методов визуализации;

2) включение в дополнение к стандартным методам визуализации (линейные графики, гистограммы, круговые диаграммы, гистограммы, диаграммы рассеяния и пр.) методов визуализации абстрактных данных, характеризующихся многомерностью и отсутствием явных пространственных привязок;

3) возможность представлять любой тип визуализации в интерактивной форме.

В результате была разработана классификация, включающая шестнадцать методов визуализации, выделенных в зависимости от цели создания визуализации (рис. 4). Эта классификация является основой для разработки онтологии языков визуализации данных, создания метамоделей этих языков.

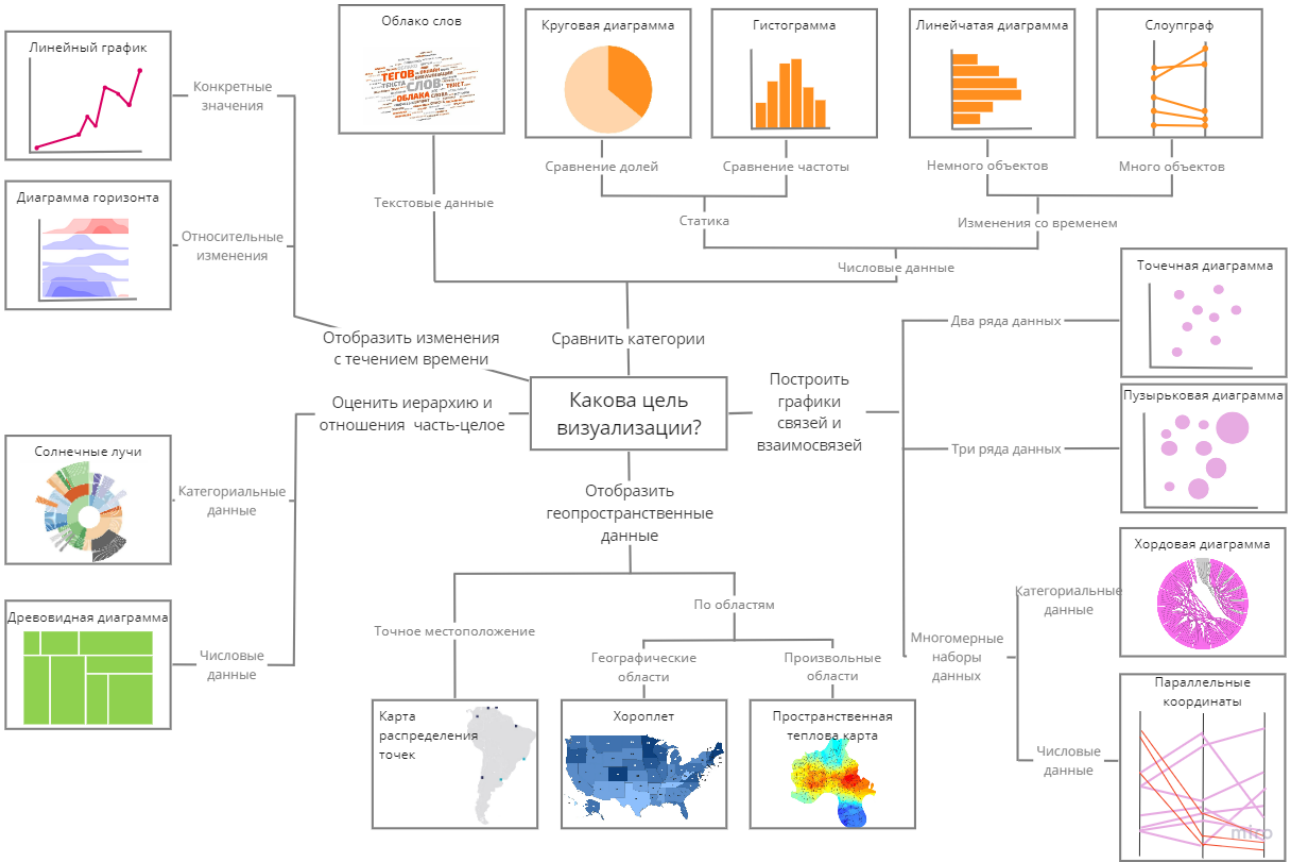


Рис. 4. Классификация методов визуализации по назначению

2. *Разработка языков для создания моделей визуализации данных*

Процесс построения онтологии языков визуализации данных включает в себя следующие этапы:

1. Формальное описание абстрактной диаграммы, которая будет включать свойства, общие для всех типов диаграмм (заголовок, легенда, ширина, высота и т. п.).
2. Выделение типов диаграмм в отдельные классы на основе разработанной классификации (рис. 4).

3. Добавление в описания классов диаграмм специфичных для каждого класса элементов диаграмм.
4. Определение отношений между классами.

Для каждого типа диаграмм формируется своя формализованная модель – представление диаграммы в онтологии, для которого создаётся DSL, позволяющий описывать этот тип визуализации.

На рис. 5 показан класс абстрактной диаграммы и его потомки в онтологии. Каждая из диаграмм состоит из отдельных элементов, имеющих свои свойства, которые также представляются в онтологии (рис. 6).

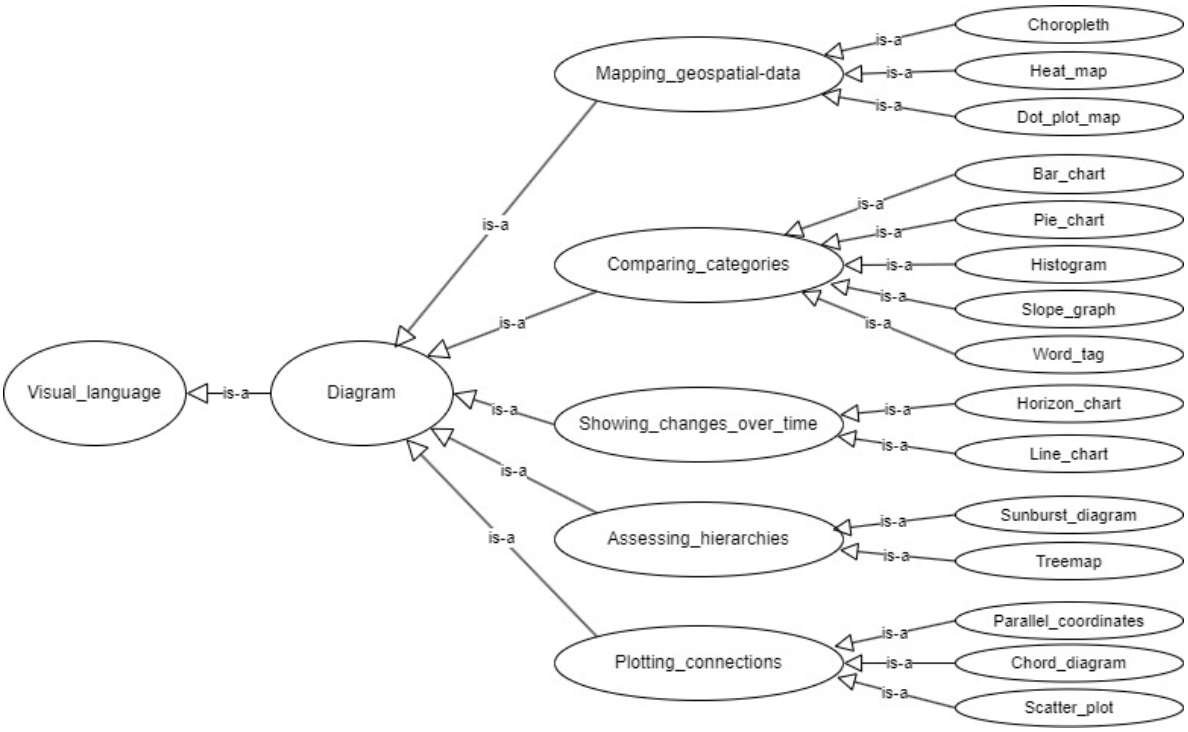


Рис. 5. Фрагмент многоаспектной онтологии: иерархия классов диаграмм в онтологии

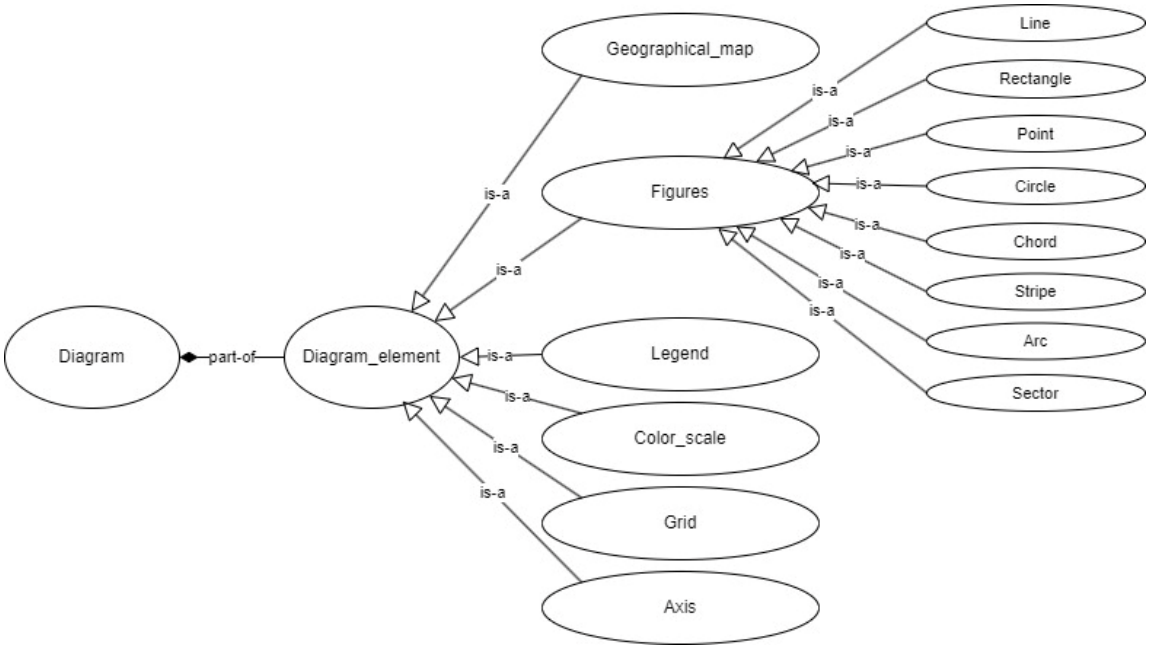


Рис. 6. Фрагмент многоаспектной онтологии: иерархия классов элементов диаграммы в онтологии

После определения основных классов и их уникальных элементов необходимо определить иерархические отношения и отношения «часть-целое» между классами. Отношения «as-is» автоматически создаются между родительским классом и дочерним классом в иерархии. Все элементы имеют собственные наборы атрибутов.

Для отображения отношений между конкретной диаграммой и ее элементами создаются специальные связи.

Для каждого типа диаграмм разрабатывается свой базовый язык. Адаптация базового языка к потребностям пользователей выполняется сопоставлением элементов диаграмм, представленных в метамодели базового языка, с элементами в описании онтологии соответствующей предметной области. На рис. 7 показана метамодель языка (DSL), который настроен на визуализацию рейтингов.

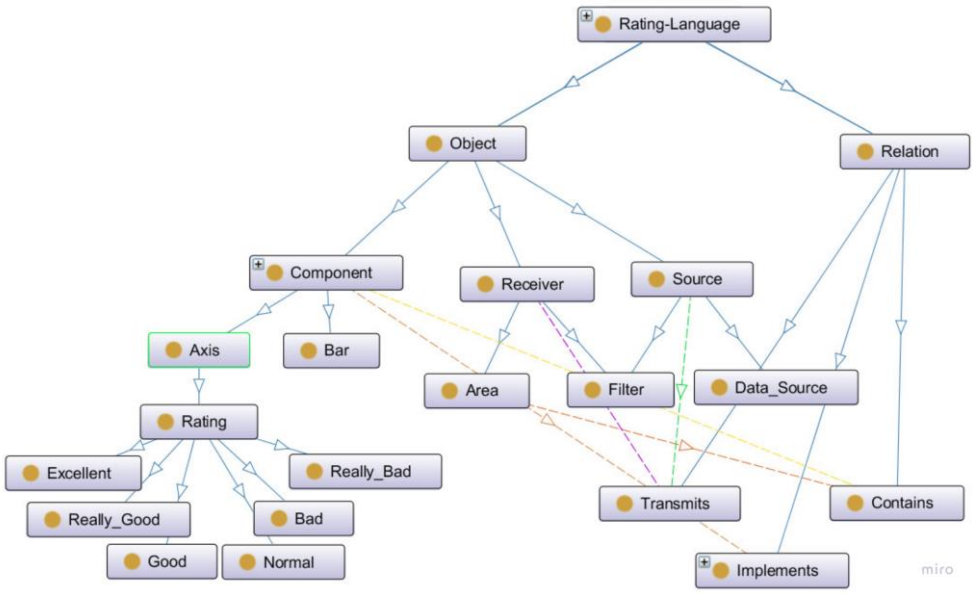


Рис. 7. Метамодель языка «Rating-Language» в онтологии визуальных языков

В разработанной метамодели DSL представлены базовые компоненты разработанной графической нотации, такие как «Axis» (оси), «Bar» (столбец), а также «Area» (область). У каждого из компонентов есть свойства (атрибуты), описание которых опущено, чтобы не загромождать иллюстрацию. С классом оси связан класс «Rating», используемый для отображения области значения допустимых рейтингов, в данном случае «Excellent», «Really_Good», «Good», «Normal», «Bad» и «Readlly_Bad». Такие классы как «Source» и «Receiver» используются для представления источника и приёмника данных соответственно. Можно заметить, что оба класса связаны с другим классом «Filter», позволяющим фильтровать данные.

С помощью описанного языка пользователь может построить свою модель визуализации, разработать диаграмму в визуальном конструкторе, определив значения всех атрибутов, указанных в метамодели языка. На рис. 8 показана диаграмма, разработанная с использованием описанного DSL.

Рис. 8. Диаграмма, построенная с использованием языка визуализации рейтингов «Rating-Language»

3. Трансформация моделей – генерация кода, реализующего пользовательскую визуализацию

С помощью разработанных DSL строятся модели визуализации, соответствующие потребностям пользователей, но для реализации разработанной модели визуализации данных требуется либо генерация кода, либо интерпретация моделей.

Подавляющее большинство современных DSM-платформ использует подход к генерации кода на основе шаблонов, что позволяет эффективно повторно использовать шаблоны [25]. Шаблоны описываются не для конкретной модели, а для метамодели [26] (в данном случае метамодели языка визуализации). Каждый шаблон состоит из двух частей – статической и динамической. Статическая часть одинакова для всех моделей, разработанных с использованием языка, определяемого метамоделью, а динамическая использует информацию, «извлеченную» из конкретной модели.

В данной работе мы предлагаем использовать подход, описанный в [21], заключающийся в создании правил трансформации моделей в визуальной среде (в конструкторе правил трансформации). Каждое правило состоит из левой части, в которой указаны объекты метамодели визуального языка, и соответствующей правой части, представляющей конструкции текстовых языков, которые и являются шаблонами.

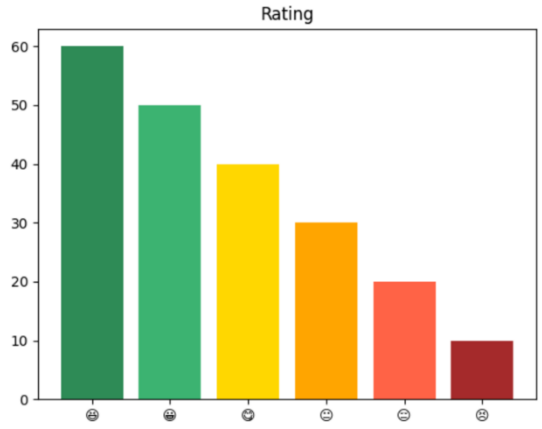
Языки Python и R являются предпочтительными языками программирования для целей визуализации данных. Python широко используется благодаря множеству подходящих библиотек, включая Matplotlib, Seaborn и Plotly, а также простому синтаксису. Хотя R уступает Python, он также обладает богатым арсеналом инструментов визуализации и остаётся популярным выбором в научных кругах. Таким образом, в качестве конструкций текстового языка в шаблонах имеет смысл использовать фрагменты кода на Python или R, содержащие вызовы библиотечных функций для визуализации данных.

После создания правил трансформации их можно применять к конкретным моделям для генерации кода пользовательской визуализации данных. Алгоритм генерации кода основан на обходе внутреннего представления моделей в виде графа.

Применив разработанные правила, можно получить программный код на соответствующем языке программирования для визуализации по правилам, заданным пользователем, разработавшим модель. Если правила трансформации разработаны для базового языка, они будут применимы ко всем моделям, разработанным с использованием всех DSL, созданных на основе этого базового языка. Это снижает трудоёмкость разработки, упрощает работу пользователей.

В качестве примера приведём код функции для построения гистограммы в Matplotlib, сгенерированный на основе приведённой выше модели по правилам трансформации, заданным пользователем в конструкторе правил трансформации, и представленным в онтологии, и результат его выполнения – диаграмму (рис. 9).

```
import matplotlib.pyplot as plt

groups = ['😆', '😀', '😋', '😐', '😒', '😣']
values = [60, 50, 40, 30, 20, 10]
colors = ['seagreen', 'mediumseagreen', 'gold',
'orange', 'tomato', 'brown']

plt.bar(groups, values, color=colors)

plt.title('Rating')

plt.show()
```



а *б*

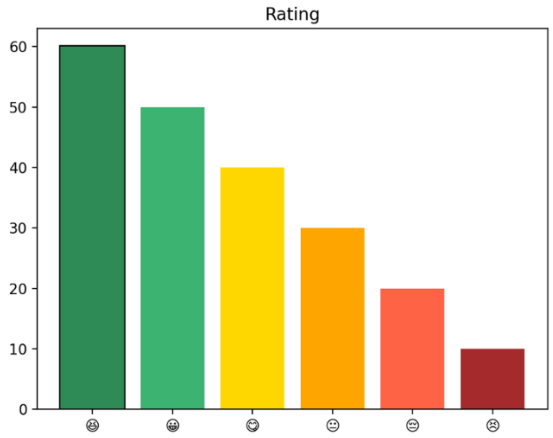
Рис. 9. Сгенерированный по правилам трансформации код (*а*) и результат его выполнения (*б*)

4. *Трансформация моделей и создание интерактивных визуализаций*

Если для языка, разработанного пользователем, описаны также и правила интерпретации, которые могут содержать правила взаимодействия пользователей с построенными визуализациями, то и код, генерируемый в ходе трансформации «Модель–Текст», будет поддерживать описанные правила интерпретации, визуализация становится интерактивной. Таким образом, при добавлении правил интерпретации, правила трансформаций автоматически расширяются данными возможностями без необходимости повторного описания правил трансформации пользователем DSM-платформы.

Код, сгенерированный для того же правила трансформации с учётом правила интерпретации, согласно которому при наведении мыши на конкретный столбец он должен выделяться, представлен на рис. 10 (*а*), а результат выполнения сгенерированного кода – на рис. 10 (*б*). Как видно из листинга, сгенерированный код действительно стал сложнее и теперь включает обработку события наведения мыши на столбцы гистограммы посредством функции «on_hover». При наведении на столбец его прозрачность уменьшится, а также его граница будет покрашена в чёрный цвет.

```
%matplotlib notebook
import matplotlib.pyplot as plt

groups = ['😆', '😳', '😋', '😳', '😔',
'😣']
values = [60, 50, 40, 30, 20, 10]
colors = ['seagreen', 'mediumseagreen',
'gold', 'orange', 'tomato', 'brown']

fig, ax = plt.subplots()
bars = ax.bar(groups, values)

def on_hover(event):
    for bar in bars:
        if bar.contains(event)[0]:
            bar.set_alpha(1.0)
            bar.set_edgecolor('black')
            bar.set_linewidth(2)
        else:
            bar.set_alpha(0.8)
            bar.set_edgecolor('none')
            bar.set_linewidth(1)
    fig.canvas.draw_idle()

fig.canvas.mpl_connect('motion_notify_event',
on_hover)

plt.bar(groups, values, color=colors)

plt.title('Rating')

plt.show()
```



а *б*

Рис. 9. Сгенерированный по правилам трансформации код интерактивной визуализации (*а*) и результат его выполнения (*б*)

V. Обсуждение результатов

Предложенный в статье подход к разработке инструментов визуализации данных, основанных на DSM, позволяет преодолеть ряд ограничений существующих инструментов визуализации и предоставить пользователям возможность настраивать модели визуализации данных для разных предметных областей за счет создания специальных языков (DSL), не предъявляя при этом высоких требований к компетенциям пользователей в области программирования.

Возможность переиспользования созданных языков, разработанных правил трансформаций снижает трудоёмкость создания визуализаций: все разработанные метамодели языков, правила сохраняются в онтологии и могут быть использованы многократно либо через настройку параметров моделей, определённых в языках, либо через создание новых языков на основе ранее разработанных.

Полученные результаты показывают перспективность предложенного подхода к разработке языков и моделей различного назначения.

Ранее языковой инструментарий был использован для описания графических нотаций, разработки визуальных языков, предназначенных для проектирования информационных систем – для описания структур данных, алгоритмов и пр., для разработки и генерации кода имитационных моделей.

Как показали полученные результаты, представленные в данной статье, возможности применения данного подхода могут быть шире: языковой инструментарий позволяет решать задачи эффективной визуализации данных и может быть интегрирован в системы различного назначения, в частности, может быть полезен при создании аналитических систем и т. п.

VI. Выводы и заключение

Научная новизна подхода к разработке языкового инструментария – в использования онтологии на всех этапах жизненного цикла инструментария, что обеспечивает возможность гибкой настройки на различные предметные области и задачи пользователей, расширение функциональных возможностей системы.

Задача апробации подхода при создании инструментов визуализации данных решена:

- На основе выполненного анализа методов и инструментов визуализации данных были разработаны метамодели базовых языков для разработки стандартных типов диаграмм – основы для генерации пользовательских визуализаций данных, получаемых из различных источников, с настройкой на специфику предметных областей, в которых работают пользователи.
- Для иллюстрации возможностей настройки визуализации на потребности пользователей разработаны на основе базовых языков метамодели DSL, которые включают нестандартные элементы диаграмм, а также элементы, обеспечивающие интерактивную визуализацию; для созданных DSL построены правила трансформации – генерации кода, при выполнении которого создаются пользовательские визуализации с заданными характеристиками.

Практическая применимость данного подхода показана на простых примерах, но в перспективе предполагается расширение возможностей разработанных программных средств (исследовательского прототипа), в частности, возможностей интерактивной визуализации посредством интерпретации созданных моделей. Теоретической основой для разработки этого подхода является Vega-Lite [27] – грамматика высокого уровня, которая позволяет быстро специфицировать интерактивную визуализацию данных. Кроме того, предлагается расширить и возможности генерации новых DSL – реализовать возможность интеграции нескольких языков в качестве базовых, комбинировать их возможности в новом DSL.

Инструменты визуализации могут быть расширены средствами оценки построенных визуализаций на соответствие критериями, рекомендациям, которые предложены в [6, 7], что должно помочь пользователям при разработке избежать ошибок, создавать эффективные визуализации.

Список литературы

1. Sawicki J., Burdukiewicz M. VisQualdex: a Comprehensive Guide to Good Data Visualization // Scientific Visualization. 2023. Vol. 15. Number 1. P. 127–149. DOI: 10.26583/sv.15.1.11.
2. Alyahya S. M. Evaluating Computer Interactions and Infographics Usability: Analyzing Individual's Performance through Viewing Patterns // Scientific Visualization. 2023. Vol. 15. Number 5. P. 111–135. DOI: 10.26583/sv.15.5.10.
3. Шамсутдинова Т. М. Визуализация знаний в учебном процессе // Научная визуализация. 2023. Т. 15. № 1, С. 100–111, DOI: 10.26583/sv.15.1.09.
4. Немтинов В. А., Родина А. А., Борисенко А. В. Морозов В. В., Протасова Ю. В., Немтинов К. В. Комплексное использование различных программных сред для повышения уровня визуализации и восприятия информации // Научная визуализация. 2023. Т. 15. № 2. P. 1–10. DOI: 10.26583/sv.15.2.01.
5. Исаев Р. А., Подвесовский А. Г. Когнитивная ясность графовых моделей: подход к пониманию идеи и способ выявления влияющих факторов с использованием визуального анализа // Научная визуализация. 2022. Т. 14. № 4. P. 38–51. DOI: 10.26583/sv.14.4.04.
6. Midway S. R. Principles of Effective Data Visualization // Patterns, 2020., Vol. 1. Issue 9. Article 100141. DOI: 10.1016/j.patter.2020.100141.
7. Midway S. R., Brum J. R., Robertson M. Show and tell: approaches for effective figures // Limnology and Oceanography Letters (L&O Letters). 2023. Vol. 8. Issue 2. P. 213–219. DOI: 10.1002/lol2.10288.
8. Oral E., Chawla R., Wijkstra M., Mahyar N., Dimara E. From Information to Choice: A Critical Inquiry Into Visualization Tools for Decision Making // IEEE Transactions on Visualization and Computer Graphics. 2024. Vol. 30. No. 1. P. 359–369. DOI: 10.1109/TVCG.2023.3326593.
9. Qin X., Luo Y., Tang N., Li G. Making Data Visualization More Efficient and Effective: a Survey // The VLDB Journal. 2019. Vol. 29. No. 1. P. 93–117. DOI: 10.1007/s00778-019-00588-3.
10. Morgan R., Grossmann G., Schrefl M., Stumptner M, Payne T. VizDSL: A Visual DSL for Interactive Information Visualization // Proc. of the 30th International Conference "Advanced Information Systems Engineering" (CAiSE 2018). 2018. P. 440–455. DOI: 10.1007/978-3-319-91563-0_27.
11. Cepero García M. T., Montané-Jiménez L. G. Visualization to Support Decision-Making in Cities: Advances, Technology, Challenges, and Opportunities // Proc of the 8th International Conference in Software Engineering Research and Innovation (CONISOFT). 2020. P. 198–207. DOI: 10.1109/CONISOFT50191.2020.00037.

12. Shakeel H. M., Iram S., Al-Aqrabi H., Alsboui T., Hill R. A Comprehensive State-of-the-Art Survey on Data Visualization Tools: Research Developments, Challenges and Future Domain Specific Visualization Framework // IEEE Access. 2022. Vol. 10. P. 96581–96601. DOI: 10.1109/ACCESS.2022.3205115.
13. Zayakin V. S., Lyadova L. N., Lanin V. V., Zamyatina E. B., Rabchevskiy E. A. An Ontology-Driven Approach to the Analytical Platform Development for Data-Intensive Domains // Knowledge Discovery, Knowledge Engineering and Knowledge Management. IC3K 2021. Communications in Computer and Information Science. Springer, Cham. 2023. Vol. 1718. P. 129–149. DOI: 10.1007/978-3-031-35924-8_8.
14. Zayakin V., Lyadova L., Rabchevskiy E. Design Patterns for a Knowledge-Driven Analytical Platform // Proceedings of the Institute for System Programming of the RAS (Proceedings of ISP RAS). 2022. Vol. 34. No. 2. P. 43–56. DOI: 10.15514/ISPRAS-2022-34(2)-4.
15. Smeltzer K., Erwig M., Metoyer R. A transformational Approach to Data Visualization // Proc. of the International Conference on Generative Programming: Concepts and Experiences (GPCE 2014). 2014. P. 53–62. DOI: 10.1145/2658761.2658769.
16. Smeltzer K., Erwig M. A Domain-Specific Language for Exploratory Data Visualization // Proc. of the 17th ACM SIGPLAN International Conference on Generative Programming: Concepts and Experiences (GPCE 2018). 2018. P. 1–13. DOI: 10.1145/3278122.3278138.
17. Ledur C., Griebler D., Manssour I., Fernandes L. G. A High-Level DSL for Geospatial Visualizations with Multi-core Parallelism Support // Proc. of the IEEE 41st Annual Computer Software and Applications Conference (COMPSAC). 2017, P. 298–304. DOI: 10.1109/COMPSAC.2017.18.
18. Ladenberger L., Dobrikov I., Leuschel M. An Approach for Creating Domain Specific Visualisations of CSP Models // Software Engineering and Formal Methods. SEFM 2014. Lecture Notes in Computer Science(). Vol. 8938. Springer, Cham. P. 20–35. DOI: 10.1007/978-3-319-15201-1_2.
19. Lyadova L., Sukhov A., Nureev M. An Ontology-Based Approach to the Domain Specific Languages Design // Proc. of the 15th IEEE International Conference on Application of Information and Communication Technologies (AICT2021). 2021. 6 p. DOI: 10.1109/AICT52784.2021.9620493.
20. Kulagin G., Ermakov I., Lyadova L. Ontology-Based Development of Domain-Specific Languages via Customizing Base Language // Proc. of the 16th IEEE International Conference on Application of Information and Communication Technologies (AICT2022). 2022. 6 p. DOI: 10.1109/AICT55583.2022.10013619.
21. Lyadova L., Ermakov I., Lanin V., Proskuryakov K. Approach to the Development of Ontology-Driven Language Toolkits Based on Metamodeling // Proc. of the IEEE 17th International Conference on Application of Information and Communication Technologies (AICT2023). 2023. 6 p. DOI: 10.1109/AICT59525.2023.10313152.
22. Matta N., Ermine J. L., Aubertin G., Trivin J. Y. Knowledge Capitalization with a Knowledge Engineering Approach: The Mask Method // Knowledge Management and Organizational Memories. Springer, Boston, MA. 2002. P. 17–28. DOI: 10.1007/978-1-4615-0947-9_2.
23. Zelazny G. The Say It with Charts Complete Toolkit. New York: McGraw-Hill Professional, 2006. 312 p.
24. Kirk A. Data Visualization: A Successful Design Process. Birmingham: Packt Publishing Ltd, 2012. 189 p.
25. Kahani N., Bagherzadeh M., Cordy J. Survey and Classification of Model Transformation Tools // Software & Systems Modeling. 2019. Vol. 18. P. 2361–2397. DOI: 10.1007/s10270-018-0665-6.
26. Ding J., Lu J., Wang G., Ma J., Kiritsis D., Yan Y. Code Generation Approach Supporting Complex System Modeling Based on Graph Pattern Matching // IFAC-PapersOnLine. 2022. Vol. 55. Issue 10. P. 3004–3009. DOI: 10.1016/j.ifacol.2022.10.189.
27. Satyanarayan A., Moritz D., Wongsuphasawat K, Heer J. Vega-Lite: A Grammar of Interactive Graphics // IEEE Transactions on Visualization and Computer Graphics. 2016. Vol. 23. No. 1. P. 341–350. DOI: 10.1109/TVCG.2016.2599030.