УДК 004.434

Проскуряков Кирилл Александрович

студент магистратуры, образовательная программа «Бизнес-аналитика»,
Пермский филиал НИУ ВШЭ,
г. Пермь
E-mail: k.proskuryakov22@gmail.com

Лядова Людмила Николаевна

доцент кафедры математического обеспечения вычислительных систем,
Пермский государственный национальный исследовательский университет,
г. Пермь
E-mail: LNLyadova@gmail.com

ЯЗЫКОВО-ОРИЕНТИРОВАННЫЙ ПОДХОД К РАЗРАБОТКЕ СРЕДСТВ ВИЗУАЛИЗАЦИИ ДАННЫХ: ГЕНЕРАЦИЯ КОДА ДЛЯ ИНТЕРАКТИВНОЙ ВИЗУАЛИЗАЦИИ

Proskuryakov Kirill Aleksandrovich

Student of the master's educational program "Business Analytics"
HSE University, Perm City
E-mail: k.proskuryakov22@gmail.com

Lyadova Lyudmila Nickolaevna

Associate Professor of the Department of Computer Systems Software
Perm State University, Perm City
E-mail: LNLyadova@gmail.com

A LANGUAGE-ORIENTED APPROACH TO DEVELOPING DATA VISUALIZATION TOOLS: GENERATING CODE FOR INTERACTIVE VISUALIZATION

Аннотация: Цель проекта – апробация подхода к генерации кода, реализующего пользовательские модели визуализации данных, на основе метамоделей визуальных предметно-ориентированных языков (DSL), созданных для описания моделей визуализации, и описаний формальных грамматик целевых текстовых языков, представленных в многоаспектной онтологии. Онтология включает также и описания правил трансформации типа «Модель-Текст» (генерации кода). Эти правила могут быть расширены для создания интерактивных визуализаций. Представлены примеры генерации кода для различных типов диаграмм, а также расширения кода на основе правил, позволяющих реализовать взаимодействие пользователя с разработанной им визуализацией. Примеры показывают

возможности представленного подхода для разработки новых моделей визуализации и их программной реализации.

Abstract: The goal of the project is to approve an approach to generating code implementing user data visualization models based on metamodels of visual domain-specific languages (DSL), created to describe visualization models, and descriptions of formal grammars of target textual programming languages presented in a multi-aspect ontology. The ontology also includes descriptions of “Model-Text” transformation rules (code generation). These rules can be extended to create interactive visualizations. Examples of code generation for various diagram types are presented, as well as rule-based code extensions that allow user interaction with the data visualization developed by them. These examples demonstrate the capabilities of the presented approach for developing new visualization models and their program implementation.

Ключевые слова: предметно-ориентированное моделирование, предметно-ориентированные языки, языковой инструментарий, многоаспектная онтология, визуализация данных, пользовательская визуализация, интерактивная визуализация, трансформация моделей, генерация кода.

Keywords: domain specific modeling, domain specific language, language toolkits, multifaceted ontology, data visualization, custom visualization, interactive visualization, model transformation, code generation.

Введение

Вопросам эффективной визуализации данных посвящены многочисленные работы, в которых подчёркивается важность разработки и использования эффективных методов визуализации [27, 7], предлагаются критерии оценивания используемых методов [21, 22], описываются подходы к пониманию идеи когнитивной ясности графовых, визуальных моделей [3].

Интерес к этой тематике связан с широким использованием визуализации как метода анализа в различных предметных областях (география и картография, медицина и образование, транспорт и энергетика, финансы и социология и др.). Визуализация данных помогает пользователям в интерпретации данных и результатов их анализа, позволяя представить их в наглядной форме. Однако эффективность работы аналитиков во многом определяется качеством визуализации данных – запутанная, сложная визуализация может привести к ошибочным выводам, неверной интерпретации результатов. В областях с интенсивным использованием данных особенно важно грамотно представить данные, показать связи между ними, а также позволить пользователям работать как с обобщёнными данными, так и с их детализацией, переходя от визуализации на одном уровне абстракции к визуализации на другом, позволяющем увидеть скрытые детали и закономерности. Эту задачу можно решить, разработав модели интерактивной визуализации данных, отвечающие потребностям пользователей.

Пользователи нуждаются в создании моделей визуализации данных, адаптированных к особенностям решаемых ими задач и предметных областей, в которых они работают. Это объясняет необходимость внедрения пользовательских спецификаций визуализации в используемые средства визуализации данных. Эти спецификации определяют правила, следуя которым пользователи могут формализовать свои требования для создания визуализаций [25]. В частности, существует потребность в быстром и простом создании интерактивных визуализаций [23]. Эти методы должны обеспечить возможность работы с различными типами и источниками данных [25], их интеграции и отбора необходимых данных.

Исследователи [24, 8] 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. Отсутствие глубоких знаний программирования у пользователей приводит к необходимости создания no-code или low-code платформ.

В статье [28] 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 [35], позволяет непрерывно отслеживать и анализировать эти данные, визуализировать результаты анализа.

Однако универсальные языки программирования не отражают потребности пользователей, что затрудняет разработку моделей, провоцирует ошибки, которые являются следствием *семантического разрыва* [33], возникающего в этом случае. Привлечение различных категорий специалистов к решению задач анализа и проектирования с самых ранних этапов возможно только при наличии средств, использование которых сокращает семантический разрыв, снижает требования к квалификации пользователей в области программирования.

Существуют различные подходы к устранению семантического разрыва [1, 19], в частности подход, основанный на *предметно-ориентированном моделировании* (*Domain Specific Modelling*, DSM), создании и использовании *предметно-ориентированных языков* (*Domain Specific Language*, DSL) – языков программирования или моделирования, специально созданных для решения определённых задач в конкретной предметной области. Такой подход называют языково-ориентированным – решение задач анализа процессов и систем или проектирования, разработки систем начинается с создания соответствующих языков. Трудоёмкость разработки смещается на создание специализированных систем программирования (или моделирования) – на разработку новых языков и инструментальных средств (редакторов, препроцессоров, компиляторов или интерпретаторов для этих языков). Частично эти проблемы снимаются при

использовании специального класса программного обеспечения – *DSM-платформ*, или *языковых инструментариев* [32].

Основной проблемой становится недоступность этих средств для конечных пользователей, специалистов в предметных областях: разработка языков требует знаний в области формальных языков и грамматик, навыков их использования.

Цель исследования, результаты которого представлены в статье, – апробация подхода к генерации кода на основе *многоаспектной онтологии*, описывающей *предметные области* и *языки*, предназначенные для решения различных задач пользователей – специалистов в этих областях.

Анализ требований и основные принципы создания DSM-платформы, основанной на знаниях и мета-моделировании, описаны в [5, 15, 20]. Подход к разработке генератора кода для DSM-платформы, управляемого знаниями, предложен в [6, 18].

Реализация средств визуализации данных: методы и этапы решения задачи

В работах [35, 36] было предложено использовать онтологии как часть архитектуры, как ядро аналитической платформы, управляемой знаниями.

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.

Обзор DSL для визуализации данных [30, 29, 17, 16] показал, большинство языков реализует небольшой набор диаграмм стандартных типов (гистограмм, круговых диаграмм и т. п.) или же эти языки предназначены для визуализации конкретных типов данных (например, геопространственных и т. п.).

Сравнение и оценки средств создания DSL в различных DSM-платформах приведены в [12, 31]. Однако ни одна из существующих промышленных DSM-платформ не имеет средств автоматизации разработки языков (DSL), не требующих глубоких знаний в области формальных языков и грамматик.

В большинстве работ предлагаются различные подходы к использованию ранее созданных языков и программных средств, интеграции и комбинированию существующих метамоделей [4, 19, 11]. Одно из перспективных направлений –

создание языковых инструментариев, основанных на знаниях, на использовании онтологий [2, 33, 34].

Для решения поставленных задач, снятия ограничений, выявленных в ходе анализа существующих средств, предложен подход к созданию языкового инструментария на основе использования многоаспектной онтологии [20, 15, 18]. Этот инструментарий может быть использован для создания средств эффективной визуализации данных с учётом рекомендаций, предложенных в [21, 22], обеспечивающих возможность создания пользовательских моделей визуализации (новых типов диаграмм, настроенных для решения конкретных задач пользователей в предметных областях, в которых она работают), а также возможность создания интерактивных визуализаций [23].

Создание пользовательских моделей визуализаций на основе рассматриваемого языково-ориентированного подхода разбивается на три этапа:

1. Разработка предметно-ориентированного языка (DSL), отражающего потребности пользователей.
2. Разработка предметно-ориентированных моделей визуализации с использованием созданного языка.
3. Генерация кода для реализации созданных моделей визуализации или их интерпретация.

В рамках представленного подхода подсистема *создания языков* позволяет пользователям *разрабатывать языки, описывая их метамодели*, или же *определяя правила автоматической генерации метамоделей DSL* – правила отображения моделей предметных областей на существующие базовые языки (созданные языки настраиваются на специфику предметной области) [2, 4]. Генерация кода также основана на использовании метамоделей DSL, а также описаний грамматик целевых текстовых языков, включённых в онтологию языков, и правил трансформации типа «Модель-Текст» (правил генерации кода по метелям, разработанным на визуальных DSL), которые также включаются в онтологию.

Генерация кода для визуализации данных

Большинство существующих DSM-платформ использует подход к генерации кода на основе шаблонов, что позволяет эффективно повторно использовать разработанные шаблоны [12], которые описываются не для конкретной модели, а для метамодели [9] (в данном случае рассматриваются метамодели языков визуализации данных). 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 данном случае предлагается использовать подход, описанный в [18], заключающийся в создании правил трансформации моделей в визуальной среде (в конструкторе правил трансформации). Каждое правило состоит из левой части, в которой указаны объекты, конструкции метамодели визуального DSL, и

соответствующей правой части, представляющей конструкции текстовых языков, которые и являются шаблонами.

Анализ показал, что языки программирования Python и R являются наиболее подходящими языками для целей визуализации данных: для Python существует множество библиотек (Matplotlib, Seaborn и Plotly), язык имеет простой синтаксис; язык R обладает богатым набором инструментов визуализации и популярен в научных кругах. Таким образом, в качестве конструкций текстового языка в шаблонах имеет смысл использовать фрагменты кода на Python или R, которые могут включать вызовы библиотечных функций для визуализации данных в этих языках.

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

Результат применения правил генерации – 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.

Примеры метамоделей визуальных языков, разработанных для визуализации данных с учётом специфики задач и потребностей конкретных пользователей, приведены в статье Ермакова И. Д., Лядовой Л. Н. «Языково-ориентированный подход к разработке средств визуализации данных: автоматизация разработки DSL», включённой в настоящий сборник.

Рассмотрим примеры генерации кода для визуализации данных (для построения описанных в упомянутой выше статье диаграмм).

Код функции для построения гистограммы в Matplotlib, сгенерированный на основе приведённой выше метамодели языка визуализации рейтинга (DSL «Rating-Language») по правилам трансформации, заданным пользователем в конструкторе правил трансформации, и представленным в онтологии, и результат его выполнения – диаграмма показаны на рис. 1 и 2.

```
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()
```

Рис. 1. Код визуализации рейтинга, сгенерированный по правилам трансформации

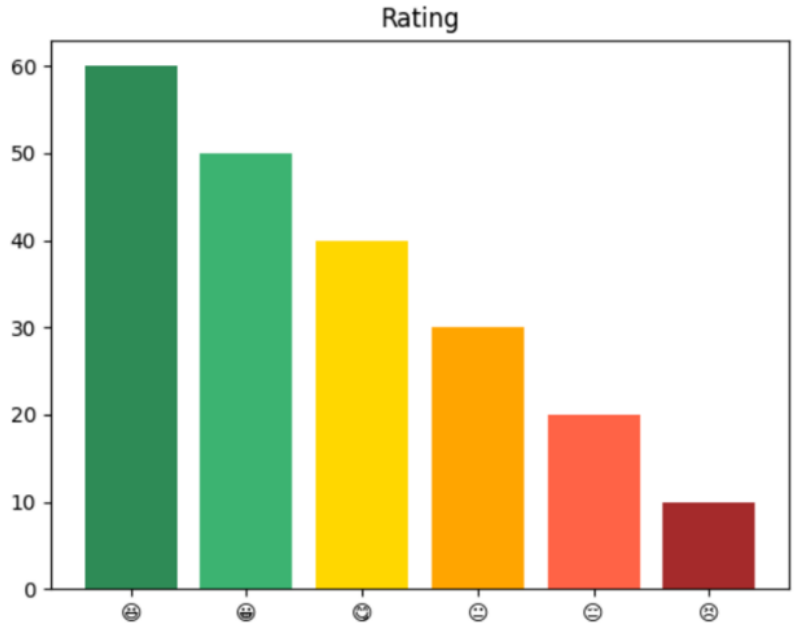

Рис. 2. Результат его выполнения сгенерированного кода визуализации рейтинга

Пользователь при разработке языка визуализации (его метамодели), может также описать правила интерпретации, правила взаимодействия пользователей с построенными визуализациями. Если описание метамодели DSL содержит такие правила, то и код, генерируемый в ходе трансформации «Модель–Текст», будет поддерживать описанные правила интерпретации, визуализация становится интерактивной. Таким образом, при добавлении правил интерпретации в разработанные метамодели языков визуализации, правила трансформаций автоматически расширяются данными возможностями без необходимости повторного описания правил трансформации пользователем DSM-платформы.

Код, сгенерированный для того же правила трансформации, которое было применено выше, с учётом правила интерпретации, согласно которому при наведении мыши на конкретный столбец он должен выделяться чёрной рамкой, представлен на рис. 3. 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.

Рассмотрим второй пример – визуализация данных о распространении видов пчел по Европе. Допустим, из имеющихся данных можно извлечь информацию не только о соотношении видов пчёл, но и о распространении конкретного вида по странам Европы. Эта информация позволяет пользователям строить новые диаграммы, детализирующие построенную. Чтобы не открывать новую диаграмму для каждого вида пчел отдельно и быстро переключаться между необходимыми данными для визуализации, в разработанной модели возможно использование объекта «Событие» (event), который позволяет определять реакцию на события, действия пользователя. Именно данный объект позволяет пользователям создавать интерактивные визуализации данных. В этом случае для выбранного элемента диаграммы (сектора), на котором установлен

курсор, предлагается построить новую диаграмму с использованием данных исходного набора данных, «содержащихся» в выбранном фрагменте. После добавления необходимого компонента и настройки дополнительной диаграммы следует интерактивно строить расширенные диаграммы, дополняющие исходные круговые диаграммы.

Результат генерации кода (применения правил преобразования для метамодели DSL с на основе круговой диаграммы) для интерактивной визуализации представлен на рис. 4. Здесь разработана интерактивная визуализация, показывающая распределение видов пчел по Европе. Вид формируемых диаграмм (круговой диаграммы и гистограммы, появляющейся при наведении курсора на сектор) показан на рис. 5.

```
%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()
```

Рис. 3. Сгенерированный код интерактивной визуализации рейтинга

```
%matplotlib notebook
import matplotlib.pyplot as plt
labels = ['Carpathian', 'Krainskaya', 'Italian']
sizes = [60, 20, 20]
colors = ['#4682b4', '#e57373', '#fdd835']
explode = (0, 0, 0)

countries = ['Slovenia', 'Austria', 'Serbia', 'Hungary', 'Romania']
values = [10, 25, 20, 30, 15]
fig, ax_pie = plt.subplots(figsize=(16, 7))
wedges, texts, autotexts = ax_pie.pie(
    sizes, explode=explode, colors=colors, autopct='%1.0f%%', startangle=90,
    textprops={'fontsize': 14}, wedgeprops={'linewidth': 1, 'edgecolor': 'white'})
ax_pie.legend(wedges, labels, loc="center left", bbox_to_anchor=(0.0, 0.5),
title="Bees")
ax_pie.set_title('Distribution of bees across Europe', fontsize=16)
ax_bar = None

def update_bar_chart():
    global ax_bar
    if ax_bar is None:
        ax_bar = fig.add_axes([0.7, 0.2, 0.25, 0.6])
    ax_bar.clear()
    ax_bar.bar(countries, values, color='skyblue', width=0.4)
    ax_bar.set_title('Distribution of Krainskaya bees by country', fontsize=12)
    ax_bar.set_xlabel('Countries')
    ax_bar.set_ylabel('Count')
    fig.canvas.draw_idle()
def hide_bar_chart():
    global ax_bar
    if ax_bar:
        ax_bar.remove()
        ax_bar = None
        fig.canvas.draw_idle()
def on_hover(event):
    found = False
    for i, wedge in enumerate(wedges):
        if wedge.contains(event)[0]:
            if labels[i] == 'Krainskaya':
                update_bar_chart()
            else:
                hide_bar_chart()
            wedge.set_alpha(1.0)
            found = True
        else:
            wedge.set_alpha(0.8)
    if not found:
        hide_bar_chart()
    fig.canvas.draw_idle()

fig.canvas.mpl_connect('motion_notify_event', on_hover)
ax_pie.axis('equal')
plt.show()
```

Рис. 4. Сгенерированный код интерактивной визуализации распределения видов пчел по Европе

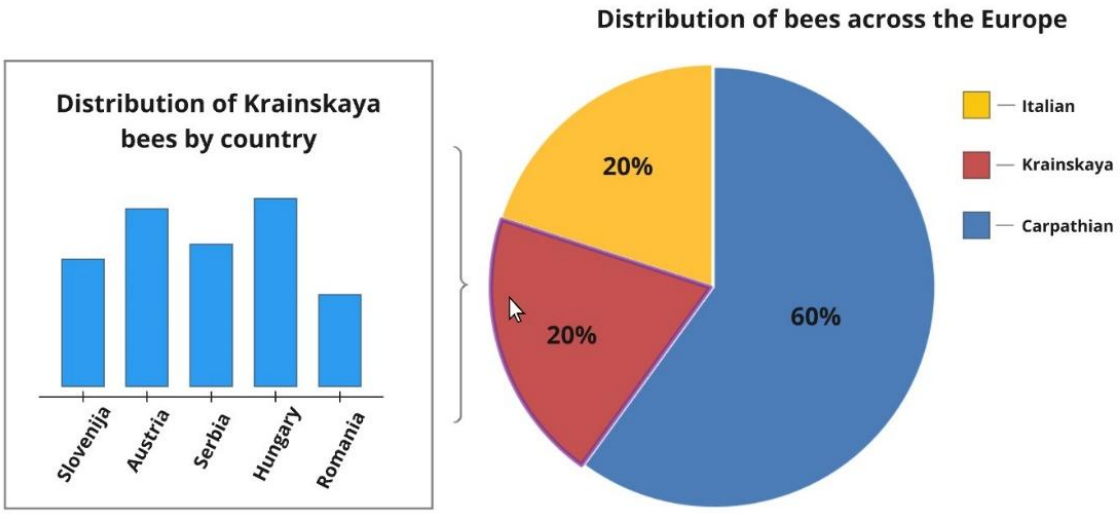

Рис. 5. Расширенная (интерактивная) круговая диаграмма, показывающая распределение видов пчел по странам Европы при наведении курсора на соответствующий сектор

Заключение

Разработанный прототип средств генерации кода для интерактивной визуализации показал, что предложенный подход имеет практическую значимость для разработки средств визуализации данных, обладающих возможностями настройки на различные предметные области и потребности пользователей.

Интерактивная визуализация обеспечивает дополнительные возможности гибкой визуализации при работе с большими массивами данных, которые могут быть сгруппированы по различным признакам и визуализированы различными способами.

Разработанные алгоритмы генерации могут быть усовершенствованы, чтобы снять ограничения на вид правил генерации для различных целевых языков.

Список литературы

1. Дегтярев А. А. Анализ способов сокращения семантического разрыва при разработке программного обеспечения / А. А. Дегтярев // Технологии разработки информационных систем: материалы конференции ТРИС-2012. Т. 1. – Таганрог: Изд во ТТИ ЮФУ, 2012. – С. 146–150.
2. Ермаков И. Д. Автоматизация разработки предметно-ориентированных языков на основе онтологии / И. Д. Ермаков // Технологии разработки информационных систем: сб. матер. XII Международной научно-технической конференции ТРИС-2022. – Таганрог: Издательство ЮФУ, 2022. С. 63–71.
3. Исаев Р. А. Когнитивная ясность графовых моделей: подход к пониманию идеи и способ выявления влияющих факторов с использованием визуального анализа / Р. А. Исаев, А. Г. Подвесовский // Научная визуализация. 2022. Т. 14. № 4. P. 38–51. DOI: 10.26583/sv.14.4.04.

4. Кулагин Г. А. Разработка иерархии предметно-ориентированных языков описания алгоритмов для устройств компании GalileoSky / Г. А. Кулагин // Технологии разработки информационных систем: сб. матер. XII Международной научно-технической конференции ТРИС-2022. – Таганрог: Издательство ЮФУ, 2022. С. 80–88.
5. Лядова Л. Н. Автоматизация разработки предметно-ориентированных языков на основе многоаспектных онтологий / Л. Н. Лядова // Информатизация и связь. 2021. № 8. С. 48–52.
6. Проскуряков К. А. Разработка DSM-платформы: средства трансформации моделей вида «модель-текст» / К. А. Проскуряков, Л. Н. Лядова // Технологии разработки информационных систем: сб. матер. XIII Международной научно-технической конференции ТРИС-2023. – Таганрог: Издательство ЮФУ, 2023. С. 124–134.
7. Alyahya S. M. Evaluating Computer Interactions and Infographics Usability: Analyzing Individual's Performance through Viewing Patterns / S. M. Alyahya // Scientific Visualization. 2023. Vol. 15. Number 5. P. 111–135. DOI: 10.26583/sv.15.5.10.
8. Cepero García M. T. Visualization to Support Decision-Making in Cities: Advances, Technology, Challenges, and Opportunities / M. T. Cepero García, L. G. Montané-Jiménez // Proc of the 8th International Conference in Software Engineering Research and Innovation (CONISOFT). 2020. P. 198–207. DOI: 10.1109/CONISOFT50191.2020.00037.
9. Ding J. Code Generation Approach Supporting Complex System Modeling Based on Graph Pattern Matching / J. Ding, J. Lu, G. Wang, J. Ma, D. Kiritsis, Y. Yan // IFAC-PapersOnLine. 2022. Vol. 55. Issue 10. P. 3004–3009. DOI: 10.1016/j.ifacol.2022.10.189.
10. Dzheiranian A. D. Designing Data Visualization System Based on Language-Oriented Approach / A. D. Dzheiranian, I. D. Ermakov, K. A. Proskuryakov, L. N. Lyadova // Proceedings of the Institute for System Programming of the RAS (Proceedings of ISP RAS). 2024. № 36(2). P. 127–140. https://doi.org/10.15514/ISPRAS-2024-36(2)-10.
11. Jácome S. Controlling Meta-Model Extensibility in Model-Driven Engineering / S. Jácome, J. De Lara // IEEE Access. – 2018. – V. 6. – P. 19923–19939. DOI: 10.1109/ACCESS.2018.2821111.
12. Kahani N. Survey and Classification of Model Transformation Tools / N. Kahani, M. Bagherzadeh, J. Cordy // Software & Systems Modeling. 2019. Vol. 18. P. 2361–2397. DOI: 10.1007/s10270-018-0665-6.
13. Kelly S. Empirical Comparison of Language Workbenches / S. Kelly // DSM'13: Proceedings of the 2013 ACM workshop on Domain-specific modeling. Indianapolis. – 2013. – P. 33–38. DOI: 10.1145/2541928.2541935.
14. Kirk A. Data Visualization: A Successful Design Process / A. Kirk. – Birmingham: Packt Publishing Ltd, 2012. 189 p.
15. Kulagin G. Ontology-Based Development of Domain-Specific Languages via Customizing Base Language / G. Kulagin, I. Ermakov, L. Lyadova // Proc. IEEE 16th International Conference on Application of Information and Communication

Technologies (AICT). – Washington : IEEE. – 2022. – P. 1–6. DOI: 10.1109/AICT55583.2022.10013619.
16. Ladenberger L. An Approach for Creating Domain Specific Visualisations of CSP Models / L. Ladenberger, I. Dobrikov, M. Leuschel // 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.
17. Ledur C. A High-Level DSL for Geospatial Visualizations with Multi-core Parallelism Support / C. Ledur, D. Griebler, I. Manssour, L. G. Fernandes // Proc. of the IEEE 41st Annual Computer Software and Applications Conference (COMPSAC). 2017, P. 298–304. DOI: 10.1109/COMPSAC.2017.18.
18. Lyadova L. Approach to the Development of Ontology-Driven Language Toolkits Based on Metamodeling / L. Lyadova, I. Ermakov, V. Lanin, K. Proskuryakov // Proc. of the IEEE 17th International Conference on Application of Information and Communication Technologies (AICT2023). 2023. 6 p. DOI: 10.1109/AICT59525.2023.10313152.
19. Lyadova L. An Ontological Approach to the Development of Analytical Platform Language Toolkits / L. Lyadova, N. Suvorov, V. Zayakin, E. Zamyatina // Proc. IEEE 16th International Conference on Application of Information and Communication Technologies (AICT). – Washington : IEEE. – 2022. – P. 1–6. DOI: 10.1109/AICT55583.2022.10013576.
20. Lyadova L. An Ontology-Based Approach to the Domain Specific Languages Design / L. Lyadova, A. Sukhov, M. Nureev // Proceedings of the 15th International Conference on Application of Information and Communication Technologies (AICT2021). Baku, Azerbaijan: IEEE. – 2021. – P. 1–6. DOI: 10.1109/AICT52784.2021.9620493.
21. Midway S. R. Principles of Effective Data Visualization / S. R. Midway // Patterns, 2020., Vol. 1. Issue 9. Article 100141. DOI: 10.1016/j.patter.2020.100141.
22. Midway S. R. Show and tell: approaches for effective figures / S. R. Midway, J. R. Brum, M. Robertson // Limnology and Oceanography Letters (L&O Letters). 2023. Vol. 8. Issue 2. P. 213–219. DOI: 10.1002/lol2.10288.
23. Morgan R. VizDSL: A Visual DSL for Interactive Information Visualization / R. Morgan, G. Grossmann, M. Schrefl, M. Stumptner, T. Payne // 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.
24. Oral E. From Information to Choice: A Critical Inquiry Into Visualization Tools for Decision Making / E. Oral, R. Chawla, M. Wijkstra, N. Mahyar, E. Dimara // IEEE Transactions on Visualization and Computer Graphics. 2024. Vol. 30. No. 1. P. 359–369. DOI: 10.1109/TVCG.2023.3326593.
25. Qin X. Making Data Visualization More Efficient and Effective: a Survey / X. Qin, Y. Luo, N. Tang, G. Li // The VLDB Journal. 2019. Vol. 29. No. 1. P. 93–117. DOI: 10.1007/s00778-019-00588-3.
26. Robert F. Model-driven Development of Complex Software: A Research Roadmap / F. Robert, R. Bernhard R. // Proceedings IEEE Transactions on

Software Engineering. – Minneapolis. – 2007. – P. 37–54. DOI: 10.1109/FOSE.2007.14.
27. Sawicki J. VisQualdex: a Comprehensive Guide to Good Data Visualization / J. Sawicki, M. Burdukiewicz // Scientific Visualization. 2023. Vol. 15. Number 1. P. 127–149. DOI: 10.26583/sv.15.1.11.
28. Shakeel H. M. A Comprehensive State-of-the-Art Survey on Data Visualization Tools: Research Developments, Challenges and Future Domain Specific Visualization Framework / H. M. Shakeel, S. Iram, H. Al-Aqrabi, T. Alsboui, R. Hill // IEEE Access. 2022. Vol. 10. P. 96581–96601. DOI: 10.1109/ACCESS.2022.3205115.
29. Smeltzer K. A Domain-Specific Language for Exploratory Data Visualization / K. Smeltzer, M. Erwig // 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.
30. Smeltzer K. A transformational Approach to Data Visualization / K. Smeltzer, M. Erwig, R. Metoyer // Proc. of the International Conference on Generative Programming: Concepts and Experiences (GPCE 2014). 2014. P. 53–62. DOI: 10.1145/2658761.2658769.
31. Tolvanen J.-P. Effort used to create domain-specific modeling languages / J. P. Tolvanen, S. Kelly // MODELS' 18: ACM/IEEE 21th International Conference on Model Driven Engineering Languages and Systems. – 2018. – P. 235–244.
32. Tolvanen J.-P. Model-driven development challenges and solutions – experiences with domain-specific modeling in industry / J.-P. Tolvanen, S. Kelly // Proceedings of the 4th International Conference on Model-Driven Engineering and Software Development, Rome, Italy. – 2016. – P. 711–719.
33. Wang H. Ontology Supporting Model-Based Systems Engineering Based on a GOPPRR Approach / H. Wang // WorldCIST'19 2019. Advances in Intelligent Systems and Computing. – 2019. – V. 930. – P. 426–436. DOI: 10.1007/978-3-030-16181-1_40.
34. Yang P. A Knowledge Management Approach Supporting Model-Based Systems Engineering / P. Yang, J. Lu, L. Feng, S. Wu, G. Wang, D. Kiritsis // Trends and Applications in Information Systems and Technologies. WorldCIST 2021. Advances in Intelligent Systems and Computing. – 2021. – V. 1366. – P. 581–590. DOI: 10.1007/978-3-030-72651-5_55.
35. Zayakin V. S. An Ontology-Driven Approach to the Analytical Platform Development for Data-Intensive Domains / V. S. Zayakin, L. N. Lyadova, V. V. Lanin, E. B. Zamyatina, E. A. Rabchevskiy // 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.
36. Zayakin V. Design Patterns for a Knowledge-Driven Analytical Platform / V. Zayakin, L. Lyadova, E. Rabchevskiy // 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.