Серия «Математика»
2022. Т. 40. С. 93—103

Онлайн-доступ к журналу:
http://mathizv.isu.ru

ИЗВЕСТИЯ
Иркутского
государственного
университета

Научная статья

УДК 004.5
MSC 68U35, 68N15
DOI https://doi.org/10.26516/1997-7670.2022.40.93

Low-code и объектные электронные таблицы

Д. Э. Гаврилина[1], **А. В. Манцивода**[1]✉

[1] Иркутский государственный университет, Иркутск, Российская Федерация
✉ ontobox@ontobox.ru

Аннотация. В работе рассматривается технология low-code разработки приложений, основанная на идее визуализации объектно-ориентированного программирования (ООП). В качестве ядра используется разработанная нами версия ООП, позволяющая управлять объектными моделями как микросервисами. Описывается метод построения визуальных инструментов управления объектными моделями как основы low-code разработки. Предлагается подход к визуализации на основе так называемых объектных электронных таблиц. Обсуждается методология использования электронных таблиц для работы с объектными структурами, её реализация в рамках платформы Ontobox, а также результаты апробации данного интерфейса к объектным моделям.

Ключевые слова: low-code, объектные электронные таблицы, микросервисное объектно-ориентированное программирование, Ontobox

Ссылка для цитирования: Гаврилина Д. Э., Манцивода А. В. Low-code и объектные электронные таблицы // Известия Иркутского государственного университета. Серия Математика. 2022. Т. 40. С. 93–103.
https://doi.org/10.26516/1997-7670. 2022.40.93

Research article

Low-Code and Object Spreadsheet

Daria E. Gavrilina[1], **Andrei V. Mantsivoda**[1]✉

[1] Irkutsk State University, Irkutsk, Russian Federation
✉ ontobox@ontobox.ru

Abstract. The paper considers low-code application development technology based on the idea of object-oriented programming (OOP) visualization. As the core, we use a version of OOP developed by us, which supports managing object models as microservices. The paper considers visual tools for managing object models as the basis for low-code development. An approach to visualization based on the so-called object spreadsheets is proposed. The spreadsheet methodology for object structure management, its implementation within the Ontobox platform, as well as the results of testing this interface to object models are discussed.

Keywords: low-code, object sreadsheet, microservice object-oriented programming, Ontobox

For citation: Gavrilina D. E., Mantsivoda A. V. Low-Code and Object Spreadsheet. *The Bulletin of Irkutsk State University. Series Mathematics*, 2022, vol. 40, pp. 93–103. (in Russian) https://doi.org/10.26516/1997-7670.2022.40.93

1. Введение

В работе рассматривается технология объектно-ориентированного low-code (OOLC) — подход к low-code разработке приложений, основанный на идее визуализации объектно-ориентированного программирования (ООП). В качестве ядра технологии используется разработанная нами версия ООП, базирующаяся на концепции документного моделирования [1]. Эта концепция позволяет расширить базовые возможности объектных моделей дополнительными инструментами, включая нативное моделирование бизнес-процессов и управление жизненным циклом объектов. С информационной точки зрения объектные модели в данном подходе обогащаются свойствами, присущими микросервисам [10], поэтому данный вариант ООП назван нами *микросервисным* объектно-ориентированным программированием.

Технология OOLC принципиально отличается от наиболее популярной сегодня low-code технологии, основанной на диаграммах описания бизнес-процессов [4], и освобождена от ряда её тяжелых недостатков [11]. OOLC реализована в рамках платформы Ontobox (ранее — платформа bSystem), апробирована и в настоящее время активно используется для решения прикладных проблем.

Данная статья сфокусирована на задаче визуализации процесса управления объектными моделями. Визуализация играет ключевую роль в low-code, заменяя текстовое кодирование. Нами разработан подход к визуализации на основе так называемых объектных электронных таблиц, позволяющий работать с объектными моделями в «Excel-стиле». Обсуждается методология использования объектных электронных таблиц, её реализация в рамках платформы Ontobox, а также результаты апробации данного интерфейса к объектным моделям.

2. Low-code: преимущества и проблемы развития

Развитие low-code платформ для разработки приложений сегодня является одним из ключевых трендов в IT-технологиях. Low-code — это подход к созданию программных систем, который не требует либо требует в минимальной степени, текстового кодирования. Вместо этого используются визуальные средства (диаграммы, таблицы и т. д.). Low-code рассматривается как одно из наиболее перспективных средств, направленных на решение массовых задач цифровизации и смягчение дефицита разработчиков, который начинает принимать критические формы. Инновационные low-code технологии способны ускорить разработку в 10–20 раз и существенно снизить порог сложности для непрофессионалов. Эти прорывные возможности обеспечили впечатляющий рост отрасли и превратили ряд low-code компаний в «единорогов» с мультимиллиардной капитализацией (например, [2; 9]).

К сожалению, до сих пор low-code платформы предлагают довольно скудный спектр возможностей, позволяют решать лишь ограниченный круг задач, а также сталкиваются с рядом серьезных технологических проблем. Критики (во многом это профессиональные разработчики, см., например, [3; 11]) фокусируются на проблемах модульности, масштабируемости, лоскутности решений, низкой производительности, безопасности и надежности, сложности при обработке специфических ситуаций и управлении крупными проектами.

С нашей точки зрения, корень проблем low-code кроется в том, что он развивается как отдельная изолированная технология. Классическое программирование (high-code) умеет решать перечисленные выше проблемы (что потребовало многолетних усилий). К сожалению, в силу своей изолированности low-code не может опереться на эти достижения, и поэтому low-code платформы вынуждены брать на себя ответственность за весь спектр этих вопросов.

Второй и не менее важный круг проблем связан с алгоритмической ограниченностью используемых сегодня подходов. Как правило, low-code технологии базируются на интерпретации («вычислении») диаграмм бизнес-процессов – вариаций спецификации BPMN [4]. Каждая платформа, в зависимости от круга решаемых задач, формирует свой язык диаграмм с собственной моделью данных и процедурной семантикой. Обычно такой язык далек от полноты, и даже если задача решается, интерпретация диаграмм часто недостаточно эффективна для работы в реальном времени. Это существенно сужает круг задач, реализуемых в чистом low-code стиле. Приходится задействовать традиционное программирование, что существенно усложняет проект и заставляет решать проблемы интеграции low-code и high-code с несовместимыми моделями данных и процедурной семантикой.

BPMN нотация (Business Process Management Notation), версии которой традиционно используются в качестве визуальной основы для low-code, также добавляет ряд проблем:

- диаграммы BPMN задают процедурные описания, которые затеняют декларативную, логическую составляющую. Это осложняет построение семантически нагруженных масштабных проектов;
- управление сложными проектами с большим количеством диаграмм — крайне непростая задача;
- отсутствует стандартизированное понимание семантики диаграмм, поэтому приложения заперты внутри платформ;
- имеются сложности с обучением непрофессионалов из-за необходимости формировать элементы алгоритмического мышления. В исследовании [7] проблема подготовки называется основным препятствием к использованию low-code в компаниях.

С нашей точки зрения, указанные проблемы исходят из самой природы диаграмм бизнес-процессов и не могут быть решены в рамках имеющейся технологии. Требуются кардинальные шаги, а именно отказ от диаграмм как основы low-code. Естественным решением является «переворачивание пирамиды»: не брать за основу визуальные средства в виде диаграмм с их дальнейшим превращением в подобие программирования, а опереться на традиционное программирование и, надстраивая над ним средства визуализации, превратить его в low-code. Эта идея реализуется в нашем проекте Ontobox, где в качестве основы взято объектно-ориентированное программирование (ООП).

3. Микросервисное ООП

Для реализации технологии объектно-ориентированного low-code (OOLC) было необходимо решить двуединую задачу:

1) адаптировать ООП к нуждам low-code;

2) надстроить над данной версией ООП инструменты визуализации.

Очевидно, что новая технология имеет смысл только тогда, когда она не уступает по своим качествам традиционному low-code, а по ряду ключевых индикаторов превосходит его. Это было для нас основным критерием разработки.

Первый этап — адаптация ООП — был обязателен, поскольку, как оказалось, задача построения low-code предъявляет к среде ООП ряд жестких требований. Прежде всего нам пришлось развить понятие объектной модели, поскольку объектные модели в традиционном понимании (например, как в языках Java, Scala, C++) не могли предоставить

нужные возможности. Расширенная версия объектных моделей (назовем их OOLC-моделями) обогащает традиционные модели по следующим направлениям: OOLC-модели долговременные (СУБД), на них определен навигационный язык запросов. Кроме того, добавлены инструменты моделирования бизнес-процессов и управления жизненным циклом объектов. Также добавлена возможность определять на модели программные интерфейсы (API) для удаленной работы с ней. С точки зрения информационной среды эти добавленные качества превращают объектную модель в микросервис [10].

Такая модернизация ООП потребовала полного контроля над языком программирования и его реализацией. Поэтому нами был развит Libretto — собственный компилируемый язык ООП со статической типизацией. Использование собственного малоизвестного языка существенно осложняет продвижение технологии, однако только благодаря ему нам удалось обеспечить сочетание необходимых свойств:

- динамическая компиляция «на лету» позволяет платформе автоматически в реальном времени перестраивать и перезапускать приложение в результате работы визуальных средств;
- Libretto работает и на сервере, и на клиенте, и в долговременных объектных моделях (в базах данных как язык запросов);
- подъязык Libretto для запросов к объектным моделям обладает строгой логической семантикой [6] — для бизнес-аналитики и ИИ;
- для упрощенного использования непрофессионалами в рамках low-code имеется синтаксически простое подмножества языка;
- с другой стороны, Libretto в целом – полноценный язык ООП для профессиональной работы на уровне high-code. В частности, сама платформа Ontobox полностью написан на Libretto.

Определение 1. *Версию ООП, основанную на OOLC-моделях назовем* микросервисным *объектно-ориентированным программированием.*

4. Объектные электронные таблицы

Второй этап формирования low-code технологии состоял в построении над микросервисным ООП визуальных инструментов разработки. Визуальные инструменты решают сразу несколько задач:

- ускоряют процесс разработки приложений примерно на порядок;
- снижают порог сложности процесса разработки и тем самым расширяют круг разработчиков;
- служат основой для коллаборации разработчиков разного уровня, когда, например, эксперт, не являющийся программистом, разрабатывает декларативную часть объектной модели (включая де-

кларативное описание методов), а профессиональный разработчик доводит функционал объектной модели до исполняемого уровня;
- повышают прозрачность приложения и возможности внешнего контроля за его разработкой и эксплуатацией — с использованием визуальных инструментов как интерфейсов для контроля;
- существенно упрощают эксплуатационную поддержку приложения, его развитие и модернизацию.

ООП хорошо подходит для визуализации, поскольку оно оперирует естественными понятиями, интуитивно понимаемыми широким кругом пользователей. Особенно это касается декларативной части объектных моделей — классов (групп объектов со схожими свойствами), самих объектов (экземпляров классов), полей (свойств, характеристик, черт).

Поля в качестве значений могут содержать (ссылки на) другие объекты. Поэтому объектные модели являются графовыми (сетевыми) структурами с вершинами-объектами и ребрами-полями. Кажется, что в такой ситуации наиболее естественным способом визуализации моделей является графовое. Однако наша весьма обширная практика показывает, что это не так. Чаще работа ведется с выборками (массивами, последовательностями, множествами) объектов. А выборки естественным образом представляются в виде таблиц (реестров), в которых по строкам идут объекты, а по столбцам — их поля/свойства. Таким образом, как это ни странно на первый взгляд, табличная работа с объектными моделями является наиболее подходящей. Еще одним преимуществом электронных таблиц является их привычность для широкого круга пользователей благодаря системе Excel.

Исходя из этих соображений, нами была разработана объектная версия электронных таблиц, позволяющая строить объектные модели и управлять ими. Это касается как задания структуры (сигнатуры) моделей, так и манипуляций с конкретными объектами. В электронных таблицах задаются и редактируются классы модели, поля, роли и статусы объектов (для моделирования бизнес процессов и контроля над жизненным циклом объектов), создаются, редактируются и удаляются конкретные объекты.

Объектные электронные таблицы являются центральным интерфейсом платформы Ontobox. Через них ведется управление функционалом модели: определением методов над классами (как правило, в low-code стиле), разработкой и подключением библиотек на Libretto (это high-code), формированием API для удаленного доступа. Также обеспечивается доступ к конструктору веб-интерфейсов, позволяющему строить над моделями веб-сервисы произвольной сложности.

При построении объектных электронных таблиц мы опирались на беспрецедентный опыт электронных таблиц Excel. Excel является по факту low-code системой, которая сыграла выдающуюся роль инструмента, позволяющего непрограммистам (например, финансистам) само-

стоятельно решать задачи и закрывать тем самым управленческие лакуны без участия IT-специалистов [8]. Объектные электронные таблицы Ontobox задают примерно такой же стиль и сложность работы. Однако в бэкграунде объектных таблиц находится не относительно примитивная модель данных, как в Excel, а семантически богатые объектные модели, позволяющие тонко описывать предметные области и решать задачи произвольной сложности.

Удивительная особенность Excel состоит в том, что один и тот же интерфейс (таблицы) используется как в процессе разработки приложения, так и его эксплуатации. Это существенно упрощает работу пользователя и принципиально отличается от low-code систем, построенных на диаграммах, поскольку они требуют отдельного конструирования веб-интерфейса. Объектные электронные таблицы Ontobox также способны служить интерфейсом по-умолчанию для практической работы с моделями, хотя в Ontobox и имеется конструктор интерфейсов, который позволяет строить веб-сервисы произвольной сложности. Более того, предусмотрена возможность автоматической генерации веб-страниц по системе электронных таблиц, определенных над моделью. Еще одно отличие: электронные таблицы Ontobox — не изолированная система решения проблем, а интерфейс к мощной объектной модели в целом (на которой, например, построена вся система управления предприятием).

Ontobox предоставляет пользователю визуальный интерфейс к объектной модели с первых минут её создания («приложение по-умолчанию»). Это позволяет не тратить время на формирование интерфейсов, если в них нет нужды, а также существенно помогает процессу разработки приложений. Ontobox поддерживает решение стандартных для электронных таблиц задач — ведение реестров, управление результатами экспериментов, управление кадрами, ресурсами, списками и т. д. В этом случае в качестве front-end достаточно использовать встроенный интерфейс объектных электронных таблиц. Методы, определенные в модели, также можно запускать из таблиц. Это обеспечивает настройку таблицы на терминологию и функционал конкретной задачи (например, в методах можно объединить несколько операций в рамках одной транзакции, добавить статистический анализ и бизнес-аналитику, генерировать печатные формы документов, привязку к месенджерам, интернету вещей, и другим источникам информации с автоматическим обновлением данных и т. д.). Все это делает работу с Ontobox доступной для самого широкого круга разработчиков и пользователей.

5. Объектно-ориентированный low-code

Объектно-ориентированная технология low-code разработки приложений определяется следующим образом:

Определение 2. *Объектно-ориентированный low-code (OOLC) — технология, основанная на сочетании микросервисного ООП и объектных электронных таблиц как визуальной надстройки над ним.*

OOLC обладает всеми преимуществами традиционного low-code. Однако в отличие от традиционного, OOLC формируется не как отдельная изолированная технология, а как визуальный интерфейс к high-code, в качестве которого выступает микросервисное ООП. Это позволяет реализовать ключевое качество OOLC, которое в традиционном low-code отсутствует: в OOLC выстроена единая среда разработки для no-code, low-code и high-code с общей моделью данных. В рамках одного приложения можно комбинировать все три уровня работы и тем самым достигать оптимального баланса скорости разработки, выразительности инструментов, гибкости и эффективности самого приложения.

No-code, low-code и high-code идеально дополняют друг друга.

No/low-code обеспечивают быстроту и визуальность разработки, доступность широкому кругу разработчиков, прозрачность и открытость к развитию. High-code дает выразительные инструменты, гибкость, эффективность и безграничность областей применения. Поэтому синергия интеграции no-code, low-code и high-code способна вывести процесс разработки на принципиально новый уровень. На рис. 1 приведена архитектура объектно-ориентированного low-code в сравнении с традиционным.

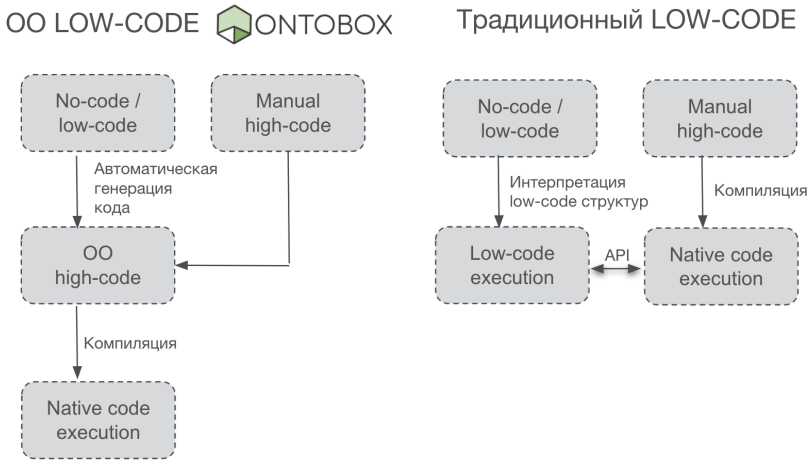

Рис. 1. Архитектуры традиционного low-code и OOLC

Единая среда формирует дуальную природу технологии OOLP и платформы Ontobox. С одной стороны, платформа может рассматриваться как интегрированная среда разработки ООП (Libretto IDE), дополненная продвинутыми визуальными инструментами управления объектными моделями. С другой стороны, это low-code платформа (low-

code application platform, LCAP [5]), которая бесшовно интегрирована с high-code средой ООП.

6. Реализация и апробация

Объектные электронные таблицы были реализованы нами в рамках платформы Ontobox как центральный интерфейс управления разработкой приложений. Ontobox была успешно апробирована на разработке ряда экспериментальных и прикладных (коммерческих) проектов, включая систему бизнес-аналитики ритейла в реальном времени (около 300 магазинов, 1 млн единичных продаж в сутки), систему управления клиентами сети мебельных магазинов, модель бюджетирования с онлайновым план-факт-контролем, систему управления регламентными работами подвижного состава ЖД, систему расчета заработной платы на основе управления мотивациями и др.

Параллельно велась оценка доступности системы для пользователей разной степени подготовленности. В частности, совместно с инновационным центром ВСЖД в декабре 2021 г. был проведен двухдневный семинар для сотрудников региональной железной дороги из разных городов, которым читалась ознакомительная лекция по Ontobox с разбором конкретного проекта (суммарно 2 часа). Участвовало 50 сотрудников ВСЖД (из них информационщиков — меньше 10%), которые затем разбились на 10 команд для формулирования и решения практических задач в рамках своей деятельности. Результат превзошел наши ожидания: 9 из 10 команд успешно справились со своей задачей с позитивной обратной связью о платформе.

В течение нескольких лет Ontobox активно применяется в учебном процессе Иркутского госуниверситета. Ontobox оказался полезным инструментом для обучения студентов (объектное моделирование, системное мышление, комплексный проектный подход). С другой стороны, от студентов была получена обширная обратная связь, повлиявшая на развитие технологии. В качестве примера: один из студентов-заочников, специализирующийся в своей работе на применении приложений 1С, решил на Ontobox некоторую задачу управления предприятием в виде веб-сервиса за 8 часов, в то время как на 1С эта же задача была решена им же за 30 часов. И это несмотря на то, что 1С предоставляет специализированную среду управления предприятием, а Ontobox — универсальная платформа разработки приложений. При этом, по его оценке, работа с Ontobox существенно более технологична и комфортна — сначала с использованием визуальных средств строится объектная модель, реализующая предметно-ориентированный язык, а затем уже в терминах этого языка решается задача.

7. Заключение

В данной работе излагается концепция объектных электронных таблиц как визуального интерфейса к объектно-ориентированному программированию и описывается методология построения low-code платформы разработки приложений, основанной на визуализации ООП. Делается обзор реализации объектных электронных таблиц в рамках платформы Ontobox.

Ontobox прошел обширную апробацию как в рамках учебного процесса, так и при решении экспериментальных и практических задач. Показано, что платформа способна повысить производительность труда разработчиков на порядок, позволяет расширить их круг за счет непрофессионалов, а также предоставляет развитые возможности для управления приложениями в процессе эксплуатации.

Список источников

1. Малых А. А., Манцивода А. В. Документное моделирование // Известия Иркутского государственного университета. Серия Математика. 2017. Т. 21. С.89–107. https://doi.org/10.26516/1997-7670.2017.21.89
2. Appian, a Low-Code Platform and Solutions. URL: https://appian.com/ (accessed 1 April 2022).
3. Arul S. IT Professionals and DevOps Say No to Low-Code // AIM. January 2022. URL: https://analyticsindiamag.com/it-professionals-and-devops-say-no-to-low-code/ (accessed 1 April 2022).
4. Business Process Model and Notation (BPMN). Version 2.0. URL: https://www.omg.org/spec/BPMN/2.0/PDF (accessed 1 April 2022).
5. Gartner Review: Magic Quadrant for Enterprise Low-Code Application Platforms. 20 Sep 2021. URL: https://www.gartner.com/doc/reprints?id=1-27I04ZJT&ct=210921&st=sb (accessed 1 April 2022).
6. Malykh A., Mantsivoda A. Query Language for Logic Architectures // Proceedings of 7th International Conference «Perspectives of System Informatics». Lecture Notes in Computer Science, 5947, Springer-Verlag, 2010. P. 294–305.
7. Martin J., Gupta S., Kumar A. Scaling with Low Code to Accelerate Digital Transformation. HFS Research & Infosys. Published November 2021. URL: https://www.infosys.com/services/digital-process-automation/insights/scaling-low-code.html (accessed 1 April 2022).
8. McCormick P. Excel Never Dies. The Spreadsheet That Launched A Million Companies. Not Boring, March 2021. URL: https://www.notboring.co/p/excel-never-dies?s=r (accessed 1 April 2022).
9. OutSystems Low-Code Development Platform. URL: https://www.outsystems.com/ (accessed 1 April 2022).
10. Wolff E. Microservices: Flexible Software Architecture. Addison-Wesley Professional, 2016. 432 p.
11. Wayner P. Why Developers Hate Low-Code. 9 Reasons Programmers Grow Frustrated with the Tools That are Supposed to Save Them Time // InfoWorld. 2019. Sep. 16. URL: https://www.infoworld.com/article/3438819/why-developers-hate-low-code.html (accessed 1 April 2022).

References

1. Malykh A.A., Mantsivoda A.V. Document Models. *The Bulletin of Irkutsk State university. Series Mathematics*, 2017, vol. 21, pp. 89-107. https://doi.org/10.26516/1997-7670.2017.21.89 (in Russian)
2. Appian, a Low-Code Platform and Solutions. Available at: https://appian.com/ (accessed 1 April 2022).
3. Arul S. IT Professionals and DevOps Say No to Low-Code. *AIM*, January 2022. Available at: https://analyticsindiamag.com/it-professionals-and-devops-say-no-to-low-code/ (accessed 1 April 2022).
4. Business Process Model and Notation (BPMN). Version 2.0. Available at: https://www.omg.org/spec/BPMN/2.0/PDF (accessed 1 April 2022).
5. Gartner Review: Magic Quadrant for Enterprise Low-Code Application Platforms. 20 Sep 2021. Available at: https://www.gartner.com/doc/reprints?id=1-27I04ZJT&ct=210921&st=sb (accessed 1 April 2022).
6. Malykh A., Mantsivoda A. Query Language for Logic Architectures // Proceedings of 7th International Conference "Perspectives of System Informatics", Springer-Verlag, Lecture Notes in Computer Science 5947, 2010, pp. 294–305.
7. Martin J., Gupta S., Kumar A. Scaling with Low Code to Accelerate Digital Transformation. HFS Research & Infosys. Published November 2021. Available at: https://www.infosys.com/services/digital-process-automation/insights/scaling-low-code.html (accessed 1 April 2022).
8. McCormick P. Excel Never Dies. The Spreadsheet That Launched A Million Companies. Not Boring, March 2021. Available at: https://www.notboring.co/p/excel-never-dies?s=r (accessed 1 April 2022).
9. OutSystems Low-Code Development Platform. Available at: https://www.outsystems.com/ (accessed 1 April 2022).
10. Wolff E. Microservices: Flexible Software Architecture. Addison-Wesley Professional, 2016, 432p.
11. Wayner P. Why Developers Hate Low-Code. 9 Reasons Programmers Grow Frustrated with the Tools That are Supposed to Save Them Time // InfoWorld, Sep 16, 2019. Available at: https://www.infoworld.com/article/3438819/why-developers-hate-low-code.html (accessed 1 April 2022).

Об авторах

Гаврилина Дарья Эдуардовна, магистрант, Иркутский государственный университет, Российская Федерация, 664003, г. Иркутск,

Манцивода Андрей Валерьевич, д-р физ.-мат. наук, проф., Иркутский государственный университет, Российская Федерация, 664003, г. Иркутск

About the authors

Daria E. Gavrilina, Postgraduate, Irkutsk State University, Irkutsk, 664003, Russian Federation

Andrei V. Mantsivoda, Dr. Sci. (Phys.-Math.), Prof., Irkutsk State University, Irkutsk, 664003, Russian Federation

Поступила в редакцию / Received 12.02.2022
Поступила после рецензирования / Revised 15.04.2022
Принята к публикации / Accepted 11.05.2021