Компьютерные инструменты в образовании, 2025
№ 2: 48–58
УДК: 004.4:005.5
http://cte.eltech.ru
doi:10.32603/2071-2340-2025-2-48-58

РАЗРАБОТКА СИСТЕМЫ АВТОМАТИЗИРОВАННОЙ ПОДГОТОВКИ ДОКУМЕНТОВ С ИСПОЛЬЗОВАНИЕМ БИБЛИОТЕКИ APACHE POI

Мамонов А. А.[1], ассистент, anton.mamonov.golohvastogo@mail.ru
Салпагаров С. И.[1], канд. физ.-мат. наук, доцент, ✉ salpagarov_si@pfur.ru, 0000-0002-5321-9650
Матюшкин Д. В.[1], магистрант, 1032212279@rudn.ru
Миронов Д. А.[1], магистрант, 1032211701@rudn.ru
Кройтор О. К.[1], канд. физ.-мат. наук, старший преподаватель, kroytor_ok@pfur.ru, 0000-0002-5691-7331

[1]Российский университет дружбы народов имени Патриса Лумумбы, ул. Миклухо-Маклая, д. 6, 117198, Москва, Россия

Аннотация

В статье рассматривается разработка программного комплекса для автоматизации документооборота, объединяющего генерацию пакетов документов на основе шаблонов и динамическое создание интерфейсов ввода. Решение реализовано на языке Java с использованием библиотеки Apache POI, оно включает обработку форматов DOC / DOCX, интеграцию с внешними данными (CSV), поддержку многопользовательских сценариев и конвертацию результатов в PDF. Система позволяет устранить ручные операции, минимизировать ошибки форматирования и повысить гибкость взаимодействия с документами. Практическое внедрение на факультете физико-математических и естественных наук РУДН продемонстрировало сокращение временных затрат на 80 % при ежемесячной обработке более 500 документов.

Ключевые слова: *автоматизация документооборота, Apache POI, генерация документов, шаблоны документов, CSV-интеграция, конвертация в PDF, динамические интерфейсы, Java, многопользовательские сценарии, обработка DOC / DOCX.*

Цитирование: Мамонов А. А., Салпагаров С. И., Матюшкин Д. В., Миронов Д. А., Кройтор О. К. Разработка системы автоматизированной подготовки документов с использованием библиотеки Apache POI // Компьютерные инструменты в образовании. 2025. № 2. С. 48–58. doi:10.32603/2071-2340-2025-2-48-58

1. ВВЕДЕНИЕ

Данная работа посвящена разработке программного комплекса для автоматизации документооборота, объединяющего генерацию пакетов документов на основе шаблонов и динамическое создание интерфейсов ввода. Решение реализовано на языке Java с использованием библиотеки Apache POI, оно включает обработку форматов DOC / DOCX, интеграцию с внешними данными (CSV), поддержку многопользовательских

сценариев и конвертацию результатов в PDF. Основной фокус направлен на устранение ручных операций, минимизацию ошибок и повышение гибкости взаимодействия с документами.

Актуальность автоматизации документооборота остается критически важной задачей для организаций, сталкивающихся с массовой генерацией стандартизированных документов (договоров, отчётов, служебных записок). Ручное заполнение шаблонов не только трудоёмко, но и приводит к ошибкам форматирования, несогласованности данных и задержкам [1]. Существующие системы, такие как ELMA или DMS-платформы, часто требуют ручной адаптации под изменяющиеся условия (например, число авторов или типы данных), сложно интегрируются с внешними источниками и ограничены в поддержке динамического контента. Разрабатываемый комплекс призван решить эти проблемы, предлагая гибкие инструменты для автоматизации, включая динамическую генерацию интерфейсов, интеграцию с CSV-таблицами и многопользовательские шаблоны [2].

Анализ существующих решений включает низкоуровневые методы (Open XML, SDK, OLE Automation, XSLT) и высокоуровневые системы (DMS, ИИ, шаблоны с динамическими полями). Однако они имеют ряд ограничений, представленных в таблице 1 [1, 3]:

- Apache POI требует глубоких технических знаний для настройки и не поддерживает автоматическую адаптацию интерфейсов.
- DMS-системы не обеспечивают динамическое изменение полей ввода в зависимости от числа авторов или структуры данных.
- Решения на основе шаблонов часто сталкиваются с проблемами интеграции с внешними источниками (CSV, SQL) и отсутствием поддержки многопоточности.

Таблица 1. Сравнение систем по ключевым критериям

Критерий	Время генерации	Сохранение форматирования	Многопользовательские шаблоны	Стоимость
Разработанная система	9,96 сек. для Huge-5500	Полное (шрифты, таблицы, стили)	Поддержка через CSV-данные, но без встроенной совместной работы	Бесплатно
Aspose.Words	7,64 сек. для Huge-5500	Полное	Требует внешних интеграций	От $1199
ELMA	Зависит от интеграции (Aspose быстрее, LibreOffice медленнее)	Возможны искажения при интеграции с LibreOffice	Встроенная поддержка (доступ для нескольких пользователей)	Бесплатная версия с ограничениями и платные лицензии

Кроме того, большинство инструментов не предоставляют механизмов для восстановления разбитых тегов или работы с разными кодировками, что критично для многоязычных сред. Текущие трудности в сфере документооборота включают:

1. Низкую гибкость шаблонов: ручная корректировка под изменяющиеся условия (число авторов, типы данных).
2. Ошибки форматирования: из-за ручного ввода и дублирования контента.
3. Сложности интеграции: несовместимость форматов данных (DOCX, CSV, SQL) и кодировок.

4. Ограниченную масштабируемость: невозможность работы с большими объемами документов в многопользовательском режиме.
5. Отсутствие стандартизации: разнородные интерфейсы для ввода данных, требующие дополнительного обучения пользователей.

Методология исследования основана на объектно-ориентированном подходе с использованием паттерна MVC. Для обработки документов применена библиотека Apache POI [4], а для хранения данных — SQLite [5]. Система автоматически генерирует интерфейс на основе шаблонов, поддерживая работу с несколькими авторами. Реализованы парсинг CSV-таблиц с обработкой кириллицы cp-1251 [6] и конвертация в PDF через documents4j [7]. Тестирование включало проверку производительности и корректности обработки данных.

2. ОПИСАНИЕ РАЗРАБОТКИ СИСТЕМЫ

Благодаря реализации программных решений для работы с документами Microsoft Office, современные системы документооборота обрели способность автоматизировать процесс создания документов на основе шаблонов. Несмотря на то, что существуют различные инструменты для работы с документами Office, такие как Microsoft Office Interop, OpenOffice API и другие, особое место среди подобных решений занимает библиотека Apache POI. Она поддерживает как файлы `.doc`, представляющие собой бинарный формат Microsoft Word [8], так и файлы `.docx`, основанные на стандарте Office Open XML [9] и представляющие собой ZIP-архив с набором XML-файлов. Помимо базовой функциональности работы с документами, Apache POI предоставляет богатый набор API для создания, чтения, редактирования и анализа различных компонентов документов, включая текст, таблицы, формулы и графические элементы. Благодаря этой универсальности и обратной совместимости, Apache POI широко используется в корпоративных системах для автоматизации документооборота, генерации отчетов и массовой обработки документов различных форматов.

В течение последних лет библиотека Apache POI активно развивалась, создавая новые возможности для оптимизации работы с документами. Это развитие включало улучшение производительности, добавление поддержки новых форматов документов и расширение функциональности для работы с макросами и формулами, что сделало библиотеку еще более мощным инструментом для разработчиков.

Результатом этого развития стала версия 5.2.5, предоставляющая широкий набор инструментов для работы с различными форматами документов. В этой версии были существенно улучшены механизмы обработки больших файлов, добавлена расширенная поддержка стилей и форматирования, а также внедрены новые API для более удобной работы с электронными таблицами и презентациями. Однако тестирование и внедрение данной версии требует создания специализированных решений, учитывающих особенности работы с различными форматами документов и спецификой их использования в реальных условиях.

На рисунке 1 можно видеть схематическое представление процесса обработки документа в разработанной системе, включающее этапы загрузки, анализа структуры, обработки содержимого документа и конвертации в PDF.

При разработке системы автоматизированной подготовки документов одним из первых этапов являлась интеграция библиотеки Apache POI через систему управления зависимостями Maven. Интеграция была реализована путем добавления соответствующих зависимостей: для работы с форматами `.docx` и `.doc` используются артефакты poi-ooxml

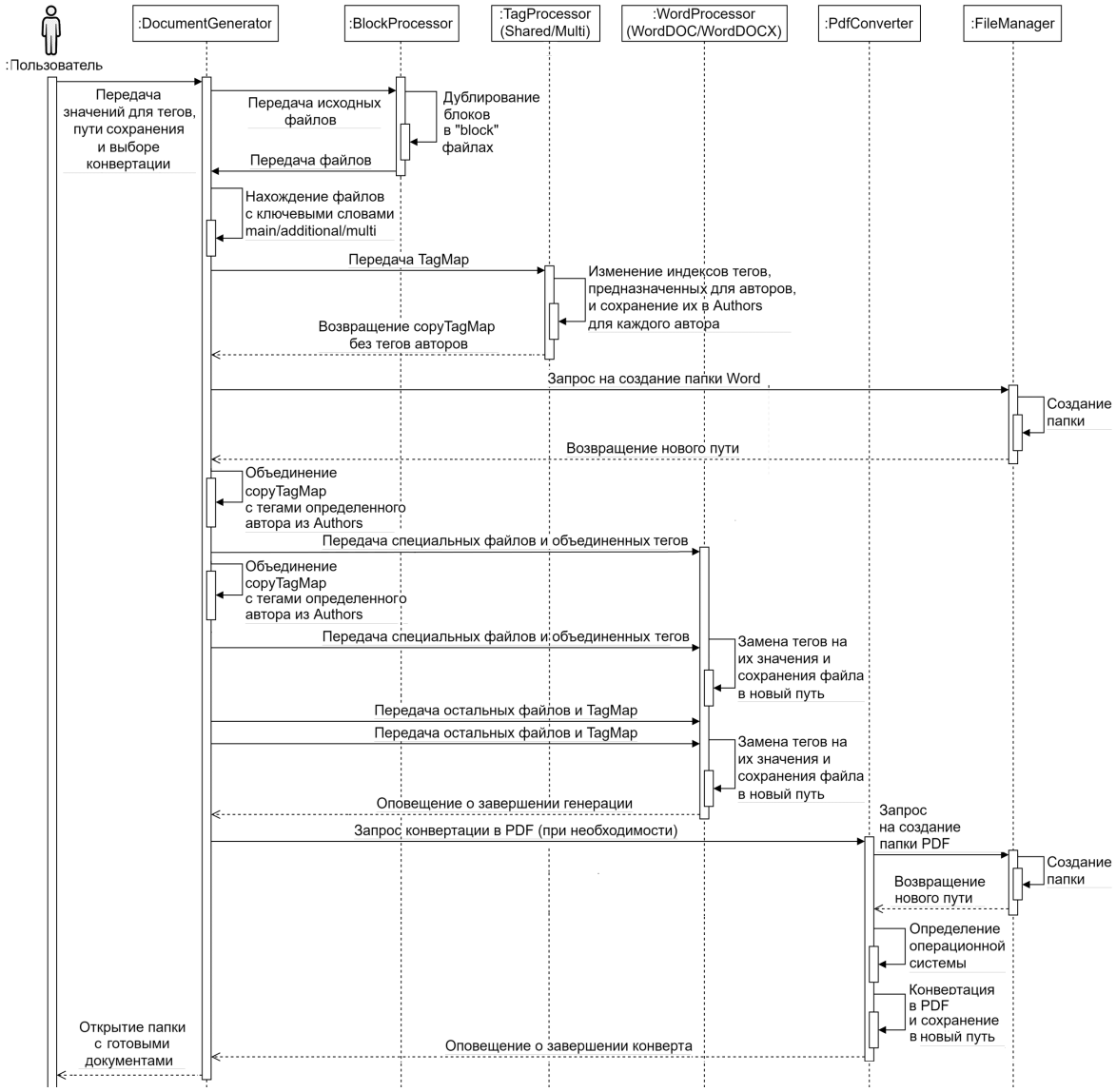

Рис. 1. Диаграмма последовательности

и poi-scratchpad соответственно. Это позволяет интегрировать библиотеку в проект и начать использование её функциональности для обработки документов.

Использование Maven в качестве системы управления зависимостями обусловлено такими его преимуществами как: автоматизированное управление зависимостями, стандартизированная структура проекта, простота развертывания и обновления компонентов. Кроме того, Maven обеспечивает централизованное хранение артефактов в репозиториях, поддерживает управление версиями, автоматическую сборку проекта и интеграцию с системами непрерывной интеграции, что значительно упрощает процесс разработки и поддержки проекта в долгосрочной перспективе.

Для реализации системы была разработана диаграмма классов (рис. 2), которая показывает внутреннюю организацию приложения, его основные сущности и взаимосвязи между ними. Данная диаграмма позволяет не только понять динамику работы системы, но и увидеть, как её компоненты организованы на структурном уровне.

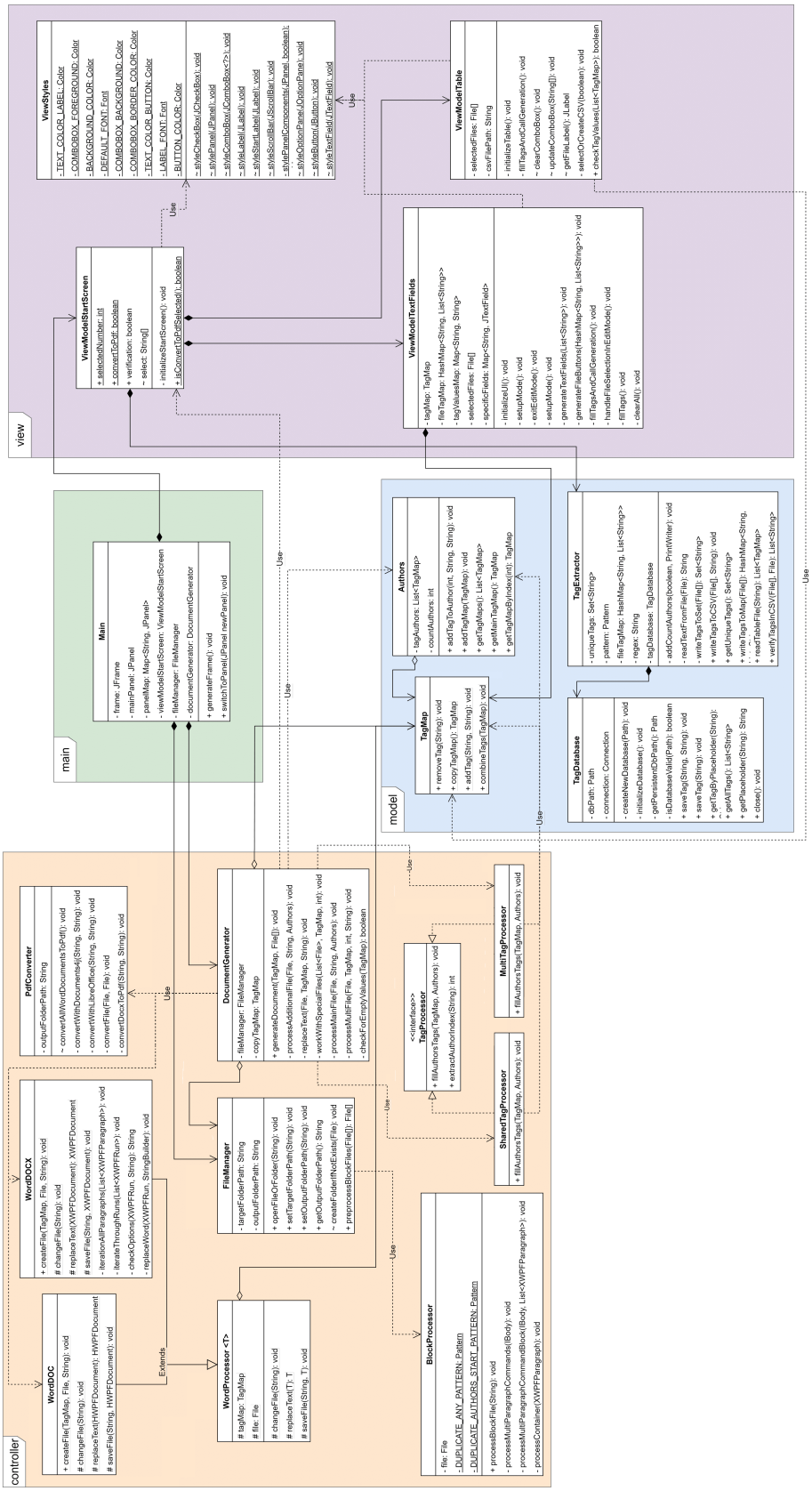

Рис. 2. Диаграмма классов

В основе такой организации лежит паттерн Model–View–Controller (MVC), обеспечивающий чёткое разделение логики, интерфейса и управления данными:

1. **Model.** Отвечает за хранение и структуру данных — в данном случае классы, связанные с хранением информации о тегах и авторах.
2. **View.** Представляет собой визуальную часть приложения (окна, формы).
3. **Controller.** Содержит основную бизнес-логику и координирует работу системы.
4. **Main.** Содержит точку входа в приложение и инициализирует контроллеры, модель и представление.

Благодаря такому распределению классов по слоям и чётким зонам ответственности упрощается поддержка и расширение приложения: новые типы тегов или форматов документов можно добавлять в соответствующие контроллеры и модели без затрагивания других компонентов.

Для обеспечения корректной обработки документов была реализована универсальная процедура обработки текстовых меток (тегов), параметризированная типом документа и способом его обработки. Границы обработки задаются структурой документа и спецификой используемого формата, что позволяет гибко конфигурировать процесс подготовки документов в соответствии с конкретными требованиями.

- **Стартовый экран** (рис. 3) позволяет задать базовые параметры документа: количество авторов, путь сохранения результатов, необходимость конвертации в PDF. Выпадающий список методов генерации (*Использовать поля ввода*, *Использовать таблицу*) определяет дальнейший workflow. Интерфейс реализует прогрессивное раскрытие функций — например, опция *Редактировать подсказки* активируется только после выбора шаблона.

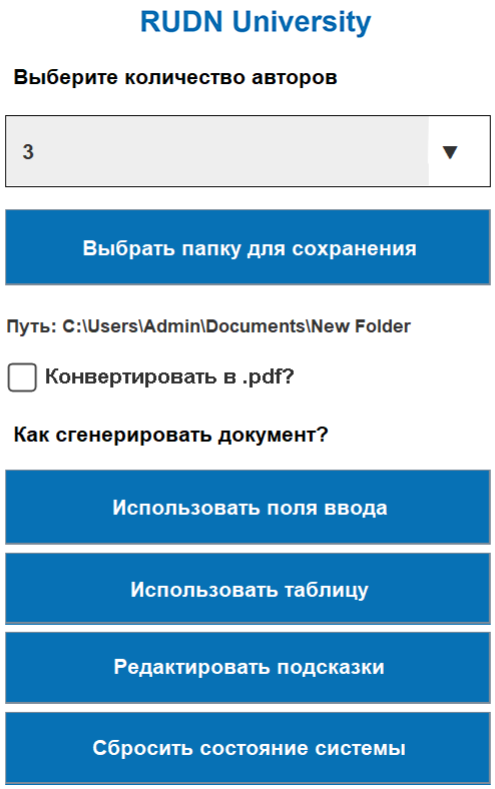

Рис. 3. Стартовое окно

- **Генерация через CSV-таблицы** (рис. 4) предоставляет инструменты для автоматизированного заполнения данных. Пользователь может создать новую таблицу, загрузить существующую или выбрать файлы шаблонов (*.doc/docx*). Поддержка динамического связывания полей (например, *Обложка для диска.docx*) с колонками CSV обеспечивает пакетную обработку документов. Алгоритм валидации предотвращает конфликт типов данных и форматирования.

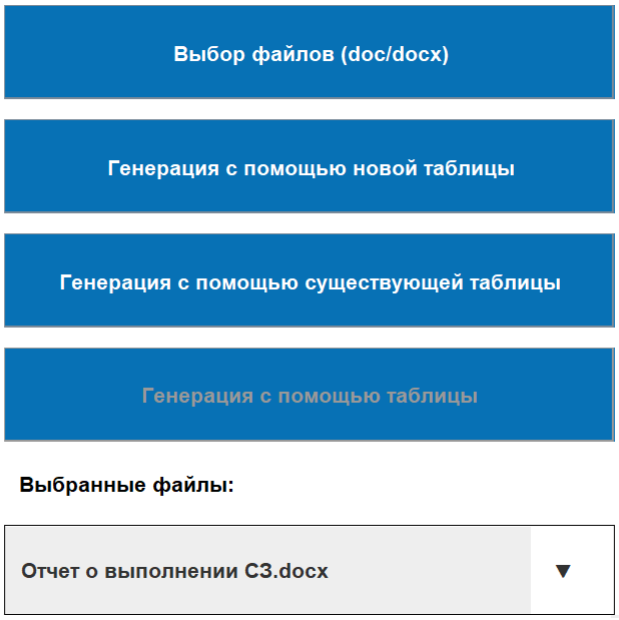

Рис. 4. Окно генерации с помощью CSV таблицы

- **Ручной ввод через текстовые поля** (рис. 5) реализует контекстно-зависимую форму с примерами заполнения (*Индекс, Страна, Город...*). Система автоматически генерирует поля на основе тегов шаблона (*$(key_ria_author1_address)*), поддерживая валидацию в реальном времени. Вкладка *Показать все теги* отображает полный список заменяемых элементов, упрощая навигацию для сложных документов.

Следует отметить, что исходные файлы данной системы размещены в открытом доступе и могут быть загружены из репозитория на платформе GitHub по адресу: https://github.com/stifell/template-process/tree/master.

3. ТЕСТИРОВАНИЕ И РЕЗУЛЬТАТЫ

Был проведен анализ времени генерации документов с использованием набора тестов, реализованных с помощью JUnit [10]. Тестирование проводилось на персональном компьютере с процессором AMD Ryzen 5 7000 Series, 16 ГБ оперативной памяти и операционной системой Windows 11. Для оценки производительности использовалась среда Java SE 21 с библиотекой Apache POI 5.2.5. Цель тестирования — изучить влияние объёма текста и количества тегов на производительность системы, а также оценить время дополнительной конвертации документов в PDF. Результаты представлены в таблице 2.

При сравнении файлов с одинаковым объёмом текста увеличение количества тегов приводит к существенному росту времени генерации. Например, для файлов с 50000 сло-

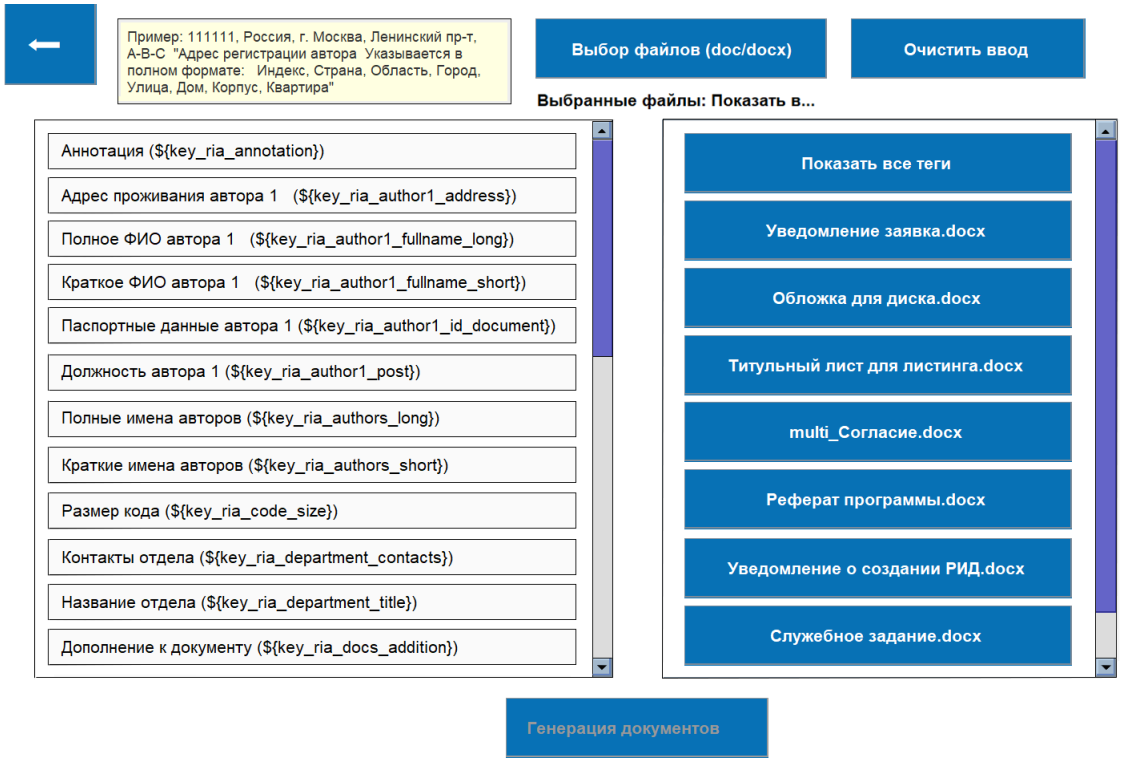

Рис. 5. Окно для заполнения документов с помощью текстовых полей

вами среднее время генерации увеличивается с 0,12864 секунд (тип Huge-1500) до 9,95829 секунд (тип Huge-5500). Аналогичная тенденция наблюдается и для файлов с 10000 словами — время возрастает с 0,09935 секунд (Medium-1500) до 1,61678 секунд (Medium-5500).

При фиксированном количестве тегов увеличение объёма текста оказывает менее заметное влияние на время генерации. Так, для файлов с 1500 тегами увеличение количества слов с 10000 (Medium-1500) до 50000 (Huge-1500) приводит к росту времени генерации лишь с 0,09935 до 0,12864 секунд. Дополнительное время, затрачиваемое на конвертацию документов в PDF, включает в себя время генерации документа. Разница между временем генерации и временем конвертации показывает, что влияние объёма текста на этот процесс более заметно, что обусловлено дополнительными операциями преобразования формата.

Таблица 2. Сводная таблица тестовых данных

Тип файла	Слов в файле	Количество файлов	Тегов на файл	Среднее время на файл (с)	Среднее время с конвертацией (с)
Small-1500	2000	100	1500	0,07305	2,83005
Medium-1500	10000	50	1500	0,09935	3,16758
Huge-1500	50000	10	1500	0,12864	7,06854
Medium-2600	10000	50	2600	0,21539	3,27142
Huge-2600	50000	10	2600	1,37381	8,51631
Medium-5500	10000	50	5500	1,61678	5,07708
Huge-5500	50000	10	5500	9,95829	17,25088

На основании проведённого тестирования можно сделать вывод, что основным фактором, определяющим время генерации документов, является именно количество тегов, внедряемых в шаблон. При этом объём текста оказывает менее значительное, но всё же ощутимое влияние.

4. ЗАКЛЮЧЕНИЕ

В результате выполнения данного проекта была разработана система автоматизированной подготовки документов на базе библиотеки Apache POI. Эта система успешно решает ключевые проблемы современных инструментов документооборота, предлагая универсальное и надежное решение. Благодаря поддержке форматов DOC и DOCX с сохранением структурной целостности, включая таблицы, формулы и графические элементы, система демонстрирует высокую адаптивность к разнородным требованиям. Интеграция с внешними источниками, такими как CSV-таблицы, с обработкой кириллических кодировок (cp-1251), обеспечивает взаимодействие с существующими корпоративными системами, устраняя необходимость ручного переноса информации.

Практическое внедрение на факультете физико-математических и естественных наук РУДН продемонстрировало сокращение временных затрат на 80 % при ежемесячной обработке более 500 документов, что подчеркивает ее готовность к использованию в реальных условиях. Особую ценность представляет многопользовательский режим, позволяющий командам совместно работать над сложными проектами без риска конфликтов версий.

Перспективы развития системы связаны с расширением функциональности через интеграцию ИИ-моделей для семантического анализа контента и автоматической генерации шаблонов, что сократит время настройки под специфические задачи. Внедрение блокчейн-технологий для верификации цепочки изменений усилит безопасность в сценариях с множеством участников, а поддержка облачных провайдеров (Google Drive, Yandex Disk) через REST API упростит распределенную работу с документами. Оптимизация кэширования для обработки экстремально больших файлов (1 млн + слов) и разработка кросс-платформенного веб-клиента сделают систему ещё более универсальной. Эти улучшения не только укрепят позиции решения на рынке, но и заложат основу для интеллектуальных систем следующего поколения, способных трансформировать подходы к управлению документами в эпоху цифровой трансформации.

Список литературы

1. *Павлов А. К.* Обзор методов автоматизации разработки документов в организации // Международный журнал гуманитарных и естественных наук. 2024. № 6–3 (93). С. 195–199. doi:10.24412/2500-1000-2024-6-3-195-199
2. *Леонова М. В.* Интерактивные интерфейсы для автоматизации документооборота. Современные технологии документооборота в бизнесе, производстве и управлении // Сборник материалов XXIV Всероссийской научно-практической конференции. Москва, 2024. М.: Абрис, 2024. С. 87–95.
3. *Кравченко В. В.* Сравнительный анализ методов генерации документов формата docx по шаблону // Математическое и программное обеспечение информационных, технических и экономических систем. Томск: Издательский Дом Томского государственного университета, 2013. С. 15–17.
4. Apache Software Foundation. Apache poi javadocs, 01 2025 [Электронный ресурс]. URL: https://poi.apache.org/ (дата обращения: 14.01.2025).

5. D. Richard Hipp. Sqlite documentation, 2023 [Электронный ресурс]. URL: https://www.sqlite.org/docs.html (дата обращения: 15.04.2025).
6. Unicode Consortium. Code page 1251 encoding, 2023 [Электронный ресурс]. URL: https://www.unicode.org/ (дата обращения: 15.04.2025).
7. documents4j Team. documents4j — document conversion api, 2023 [Электронный ресурс]. URL: https://documents4j.com/ (дата обращения: 15.04.2025).
8. Microsoft Learn. [ms-doc]: Word (.doc) binary file format [Электронный ресурс]. URL: https://learn.microsoft.com/ (дата обращения: 20.03.2025).
9. ECMA International. ECMA-376 Office Open XML File Formats, fifth edition, 2021 [Электронный ресурс]. URL: https://www.ecma-international.org/ (дата обращения: 15.03.2025).
10. JUnit Team. Junit 5 user guide [Электронный ресурс]. URL: https://junit.org/junit5/ (дата обращения: 17.04.2025).

Поступила в редакцию 19.05.2025, окончательный вариант — 15.07.2025.

Мамонов Антон Алексеевич, ассистент, Российский университет дружбы народов им. Патриса Лумумбы, anton.mamonov.golohvastogo@mail.ru

Салпагаров Солтан Исмаилович, канд. физ.-мат. наук, доцент, Российский университет дружбы народов им. Патриса Лумумбы, ✉ salpagarov_si@pfur.ru

Матюшкин Денис Владимирович, магистрант, Российский университет дружбы народов им. Патриса Лумумбы, 1032212279@rudn.ru

Миронов Дмитрий Андреевич, магистрант Российский университет дружбы народов им. Патриса Лумумбы, 1032211701@rudn.ru

Кройтор Олег Константинович, канд. физ.-мат. наук, старший преподаватель, Российский университет дружбы народов им. Патриса Лумумбы , kroytor_ok@pfur.ru

Computer tools in education, 2025
№ 2: 48–58
http://cte.eltech.ru
doi:10.32603/2071-2340-2025-2-48-58

Development of an Automated Document Preparation System Using the Apache POI Library

Mamonov A. A.[1], Assistant, anton.mamonov.golohvastogo@mail.ru
Salpagarov S. I.[1], Cand. Sc., Associate Professor, ✉ salpagarov_si@pfur.ru, 0000-0002-5321-9650
Matyushkin D. V.[1], Master's Degree student, 1032212279@rudn.ru
Mironov D. A.[1], Master's Degree student, 1032211701@rudn.ru
Kroytor O. K.[1], Cand. Sc., Senior Lecturer, kroytor_ok@pfur.ru, 0000-0002-5691-7331

[1] RUDN University, 6 Miklukho-Maklaya str., 117198, Moscow, Russia

Abstract

The article discusses the development of a software system for document workflow automation, combining the generation of document packages based on templates and dynamic creation of input interfaces. The solution is implemented in Java using the Apache

POI library, provides processing of DOC/DOCX formats, integration with external data (CSV), support for multi-user scenarios, and conversion of results to PDF. The system eliminates manual operations, minimizes formatting errors, and increases flexibility in document interaction. Practical implementation at the Faculty of Physics, Mathematics and Natural Sciences of RUDN University demonstrated an 80 % reduction in time costs when processing more than 500 documents monthly.

Keywords: *document workflow automation, Apache POI, document generation, document templates, CSV integration, PDF conversion, dynamic interfaces, Java, multi-user scenarios, DOC/DOCX processing.*

Citation: A. A. Mamonov, S. I. Salpagarov, D. V. Matyushkin, D. A. Mironov, and O. K. Kroytor, "Development of an Automated Document Preparation System Using the Apache POI Library," *Computer tools in education*, no. 2, pp. 48–58, 2025 (in Russian); doi:10.32603/2071-2340-2025-2-48-58

References

1. A. K. Pavlov, "Overview of automation methods for document development in an organization," *International Journal of Humanities and Natural Sciences*, no. 6-3 (93), pp. 195–199, 2024 (in Russian); doi:10.24412/2500-1000-2024-6-3-195-199
2. M. V. Leonova, "Interactive interfaces for document workflow automation," in *Proc. of Modern Document Management Technologies in Business, Production and Management: XXIV All-Russian Sci. Pract. Conf.*, Moscow: Abris Publishing, , pp. 87–95, 2024 (in Russian).
3. V. V. Kravchenko, "Comparative analysis of methods for generating docx format documents from a template," in *Mathematical and Software Support for Information, Technical and Economic Systems*, Tomsk, Russia: Tomsk State Univ. Press, , pp. 15–17, 2013 (in Russian).
4. "Apache Software Foundation", in *Apache POI Javadocs*, 2025. [Online]. Available: https://poi.apache.org/
5. D. R. Hipp, *SQLite Documentation*, 2023. [Online]. Available: https://www.sqlite.org/docs.html
6. "Unicode Consortium," in *Code Page 1251 Encoding*, 2023. [Online]. Available: https://www.unicode.org/
7. "documents4j Team," in *documents4j — Document Conversion API*, 2023. [Online]. Available: https://documents4j.com/
8. "Microsoft Learn," in *[MS-DOC]: Word (.doc) Binary File Format*, 2025. [Online]. Available: https://learn.microsoft.com/
9. "ECMA International," in *ECMA-376 Office Open XML File Formats*, 5th ed., 2021. [Online]. Available: https://www.ecma-international.org/
10. "JUnit Team," in *JUnit 5 User Guide*, 2025. [Online]. Available: https://junit.org/junit5/

Received 19-05-2025, the final version — 15-07-2025.

Anton Mamonov, Assistant, Friendship University of Russia (RUDN University), anton.mamonov.golohvastogo@mail.ru

Soltan Salpagarov, Cand. of Sciences (Phys.-Math.), Associate Professor, Friendship University of Russia (RUDN University), ✉ salpagarov_si@pfur.ru

Denis Matyushkin, Master's Degree student, Friendship University of Russia (RUDN University), 1032212279@rudn.ru

Dmitry Mironov, Master's Degree student, Friendship University of Russia (RUDN University) , 1032211701@rudn.ru

Oleg Kroytor, Cand. of Sciences (Phys.-Math.), Senior Lecturer, Friendship University of Russia (RUDN University), kroytor_ok@pfur.ru