Научная статья
УДК 004.912
doi: 10.17586/2713-1874-2022-3-47-57

АНАЛИЗ ВОЗМОЖНОСТЕЙ ПАРСИНГА ЭЛЕКТРОННЫХ ТЕКСТОВЫХ ДОКУМЕНТОВ ДЛЯ АВТОМАТИЗАЦИИ НОРМОКОНТРОЛЯ

***Вячеслав Игоревич Марцинкевич*[1], *Галина Сергеевна Ларионова*[2], *Владислав Витальевич Терещенко*[3], *Ксения Андреевна Ситникова*[4], *Наталия Николаевна Горлушкина*[5]✉**

[1,2,3,4,5]Университет ИТМО, Санкт-Петербург, Россия
[1]slavamarcin@yandex.ru
[2]larionovags@yandex.ru
[3]vlad-tershch@yandex.ru
[4]ksusitnikova@yandex.ru
[5]nagor.spb@mail.ru✉, https://orcid.org/0000-0002-6549-1723
Язык статьи – русский

Аннотация: В статье анализируется возможность парсинга электронных документов для организации автоматизированного контроля оформления текстовых документов на примере оформления выпускных квалификационных работ. С этой целью были определены наиболее актуальные форматы текстовых документов. Сопоставлены алгоритмы обработки DOCX-документа, PDF-документа, ODT-документа. Выяснилось, что форматы docx и odt имеют похожую структуру и способы хранения информации о контенте. В то же время формат pdf имеет совершенно другие принципы. Также собрана основная информация о структуре стилей в документах формата odt. Это – необходимый элемент для парсинга, так как стили хранят характеристики форматирования. Сравнительный анализ показал возможность парсинга всех трех текстовых форматов документов для автоматизированного контроля оформления документов. Для реализации этой цели были исследованы возможности языков Python и Java.

Ключевые слова: нормоконтроль, парсинг, стили, хранение документов, электронный текстовый документ, docx, odt, pdf, xml, Add-In technology, Microsoft Office, Python

Ссылка для цитирования: Марцинкевич В.И., Ларионова Г.С., Терещенко В.В., Ситникова К.А., Горлушкина Н.Н. Анализ возможностей парсинга электронных текстовых документов для автоматизации нормоконтроля // Экономика. Право. Инновации. 2022. № 3. С. 47–57. http://dx.doi.org/10.17586/2713-1874-2022-3-47-57.

ANALYSIS OF THE POSSIBILITIES OF PARSING ELECTRONIC TEXT DOCUMENTS FOR THE AUTOMATION OF STANDARD CONTROL

***Viacheslav I. Martsinkevich*[1], *Galina S. Larionova*[2], *Vladislav V. Tereshchenko*[3], *Kseniya A. Sitnikova*[4], *Natalia N. Gorlushkina*[5]✉**

[1,2,3,4,5]ITMO University, Saint Petersburg, Russia
[1]slavamarcin@yandex.ru
[2]larionovags@yandex.ru
[3]vlad-tershch@yandex.ru
[4]ksusitnikova@yandex.ru
[5]nagor.spb@mail.ru✉, https://orcid.org/0000-0002-6549-1723
Article in Russian

Abstract: The article analyzes the possibilities of parsing documents for organizing automated document standard control system, using the example of final qualification works. For this purpouse the most relevant formats of text documents have been determined. The features of docx, pdf, odt documents have been compared. It turned out that the docx and odt formats have a similar structure and ways of storing information about the content. At the same time, the pdf format has completely different principles for this. Basic information about the structure of styles in odt documents was collected. They are necessary elements for parsing since they store formatting characteristics. A comparative analysis showed the possibility of parsing all of these three most relevant text document formats for automated control of document processing. To achieve this goal, the possibilities of the Python and Java languages were investigated.

Keywords: document storage, electronic text document, Microsoft Office Add-In technology, parsing, Python, standard control, styles, doc, odf, pdf, xml

For citation: Martsinkevich V.I., Larionova G.S., Tereshchenko V.V., Sitnikova K.A., Gorlushkina N.N. Analysis of the Possibilities of Parsing Electronic Text Documents for the Automation of Standard Control. *Ekonomika. Pravo. Innovacii.* 2022. No. 3. pp. 47–57. (In Russ.). http://dx.doi.org/10.17586/2713-1874-2022-3-47-57.

Введение. К этапу завершения обучения в вузе выпускник должен обладать определенными профессиональными компетенциями, наработанными в рамках образовательного процесса. Для различных направлений подготовки этот список варьируется: для специальностей, связанных с программированием – это «способность управлять развитием БД (ПК-5)», а для дизайнера – «проведение предпроектных дизайнерских исследований (ПК-3)». Однако есть общие навыки, которые должен приобрести каждый выпускник. Например, подготовка отчетности (ПК-12), так как именно эта компетенция определяет его как профессионала, способного вести совместные исследования, работать с коллегами и представлять результаты проделанной научно-исследовательской работы.

Согласно статистике, в российских вузах только за 2020-й год около 558 800 студентов получили степень бакалавра, 105 400 – степень специалиста, а 185 200 – степень магистра, защитив в сумме порядка 849 тысяч дипломных работ. Получается, что среднее количество отчетов к проверке на один вуз составило 1196 (при числе высших учебных заведений, обучающих бакалавров, специалистов и магистров, равным 710 по состоянию на 2020 год) [1].

ГОСТ 7.32–2017, согласно которому предоставляется отчетность о научно-исследовательской работе, содержит более 60 отдельных пунктов в разделе правил оформления отчета [2]. Неудивительно, что в такой ситуации студенты неизбежно допускают ошибки по невнимательности. Используя статистические данные исследователей из РАНХиГС, были подготовлены 40 типовых ошибок выпускников, примерно четверть из которых составили ошибки оформления [3], что говорит о существующей проблеме в подготовке отчета студентами. При анализе 500 выпускных квалификационных работ научной группой Университета ИТМО также было выявлено, что обучающиеся допускают значительное количество ошибок при оформлении работ. В основе этой проблемы лежат две причины: неумение работать с нормативно-правовой базой и восприятие ошибок, связанных с оформлением, как несущественных [4].

Поскольку отчетность проверяется преимущественно вручную, это неизбежно повышает риск ошибок, связанных с человеческим фактором, так как при большом объеме работ на одного дипломного консультанта риск не заметить при проверке мелкие ошибки возрастает. Эта насущная проблема среди университетских научных сообществ привела к генерации множества различных решений по проверке отчетности посредством автоматизации процесса нормоконтроля.

Постановка задачи (Цель исследования). Основываясь на описанной выше проблеме, в исследовании ставится цель – определить возможности парсинга различных форматов документов и дальнейшей автоматизации процесса проверки оформления текстовых документов. Для этого необходимо сравнить три наиболее популярных формата документа – docx, pdf и odt, определить, возможен ли их парсинг программными средствами, и выделить оптимальный для этого язык.

Методы и материалы исследования.

Определение актуальных форматов документов. Для того чтобы понять текущее состояние развития подобных систем, было проведено исследование рынка на наличие программ, разработанных научными сообществами в этой области [4, 5, 6, 7, 8, 9]. Результаты исследования представлены в Таблице 1.

Таблица 1

Разработки в сфере нормоконтроля оформления документов

Год	Автор / Университет	Упомянутый стек технологий	Анализируемые форматы	Охватываемый спектр задач
2015	Камчатский государственный университет им. Витуса Беринга	Delphi, SQL	docx, doc	Автоматизация формирования титульных листов. Проверка форматирования содержания. Формирование приложения к экзаменационной ведомости.
2015	Белорусский государственный технологический университет	ASP.Net, Open XML SDK	docx	Проверка форматирования содержания.
2016	НИУ «Высшая школа экономики»	Open XML SDK, C#, ASP.Net	docx	Проверка форматирования содержания.
2018	Уральский государственный педагогический университет	C#	docx	Проверка форматирования содержания. Обучение оформлению документации.
2021	Вятский государственный университет	HTML, VBA	doc, docx, rtf	Проверка форматирования содержания.
2021	Университет ИТМО	C#, .Net, Python	docx	Проверка форматирования содержания. Обучение оформлению документации.

На основе проведенного анализа можно сделать вывод, что все решения схожи и сводятся к проверке файлов исключительно формата docx на соответствие правилам оформления ГОСТ.

Подобная тенденция выбора файлов формата docx в качестве базиса для проверки на наличие ошибок оформления легко объяснима. Это обусловлено тем, что разработанные программные средства взаимодействуют со структурой xml, которая лежит в основе формата. Структура xml-документа подразумевает, что содержимое отчета распределено внутри файла во множестве элементов, каждый из которых завернут в определенный тег, отражающий его свойства и характеристики. Из такой хорошо читаемой и однозначной структуры следует простота реализации программного обеспечения по анализу содержимого документов.

Однако важно отметить, что указанное ограничение – поддержка только форматов docx – значительно. Проведя анализ наиболее часто используемых текстовых форма-

тов, становится понятно, что файлы, созданные в среде Microsoft Word, являются далеко не единственными, используемыми в сети, а самое главное – даже не занимают первую строчку лидерства в списке трендов текстовых форматов (основанного на количестве опубликованных файлов за каждый из 2011–2021 годов). В 2014 году формат pdf охватил 79% всех публикаций, docx – 17%, odt – 0.8%, а txt/rtf – 2.9%. В 2021 году показатели pdf выросли до 90%, а docx упали на 14% достигнув отметки в 3% [10, 11]. Также следует принять во внимание, что политика РФ направлена на импортозамещение [12], что означает возможный постепенный отказ от формата docx программного обеспечения Microsoft Word, входящего в платный пакет Microsoft Office.

Таким образом, исследование показывает необходимость создания современного программного обеспечения для автоматизированной работы с несколькими популярными в научном сообществе форматами.

Для определения наиболее востребованных и актуальных форматов, необходимых для внедрения в сервис по автоматизированному нормоконтролю, была проведена сравнительная характеристика наиболее популярных текстовых форматов. Результаты представлены в Таблице 2.

Таблица 2

Сравнение актуальных текстовых форматов

Расширение	TXT	ODT	WPD	RTF
Стандартизация		ГОСТ Р ИСО/МЭК 26300-2010		
Год создания	1960-е	2011	1980	Конец 1980-х
Открытость	Открытый	Открытый	Проприетарный	Проприетарный
Назначение	Хранение текстовых документов в чистом виде, не поддерживает вставку изображений, форматирование только специальными символами	Открытый формат документов, в основе лежит язык разметки XML	Текстовый формат редактора WordPerfect	Межплатформенный формат хранения размеченных текстовых документов
Наличие бесплатного ПО	Да	Да	Частично	Частично

Основываясь на данных из таблицы и рейтинге популярности форматов документов, были определены как наиболее подходящие для создания программы автоматизированного нормоконтроля следующие три формата файлов:

1. DOCX – как основное расширение, имеющееся при создании отчета в программе Word среды Microsoft Office.

2. PDF – как самое частое расширение среды Adobe Acrobat, используемое при отправке файлов по сети для оценки преподавателями. Начиная с 2008 года – открытый формат (ISO 32000).

3. ODT – как частый аналог docx, имеющий широкий спектр возможностей при условии бесплатной работы благодаря открытому исходному коду.

Актуализация языка и библиотек для разработки системы. В рамках НИРМ Университета ИТМО в период 2019–2021 годов ведется разработка «Сервиса автоматизированного нормоконтроля документов и обучения оформлению документации». На текущий момент сервис осуществляет автоматизированную проверку оформления отчетности о НИР на соответствие ГОСТам и нормативными актам Университета ИТМО.

В основе приложения задействована клиент-серверная архитектура, причем клиентская часть базируется на технологии Microsoft Office Add-In technology, что позволяет напрямую в программе Microsoft Word, не используя внешние средства, проверять оформление. Спроектированная система работает по следующему алгоритму:

1. Аутентификация и авторизация пользователя.
2. Загрузка и сохранение документа на сервер.
3. Получение свойств параграфов документа в формате csv файла.
4. Классификация каждого из параграфов текста.
5. Повышение точности классификатора с помощью методики меток (согласование результата с пользователем).
6. Проверка документа на соответствие выбранным правилам оформления.

На этапе реализации сервиса для взаимодействия с файлами была выбрана библиотека GemBox.Document, осуществляющая работу на сервере в рамках платформы .NET с помощью языка C#. Этот компонент позволяет читать, писать, редактировать, конвертировать и печатать файлы документов из приложений .NET с помощью одного API, что было оптимальным решением для разработанного программного продукта.

Однако подобная архитектура сервиса и набор используемых библиотек позволяют взаимодействовать сервису с файлами исключительно формата docx, ввиду отсутствия инструментов для работы с форматами odt и pdf. Подобное ограничение привело к решению о модернизации системы: были проанализированы особенности структуры различных форматов текстовых документов и проведен поиск и обоснование наиболее подходящего языка программирования и соответствующих библиотек для работы с файлами различных форматов.

Сравнение форматов. Сравнение было проведено по двум пунктам: структура хранения информации в документе и способ хранения информации о контенте документа.

В форматах docx и odt информация хранится в виде архива, содержащего несколько xml файлов. XML файлы хранят в себе контент, свойства, параметры, метаданные документа и взаимосвязи между файлами. В то же время pdf документ представляет собой бинарный файл, который можно разделить на следующие четыре части: заголовок, тело, xref таблица и прицеп, каждая из которых выполняет определённую роль.

Xref таблица является одной из основных отличительных особенностей данного формата и представляет собой строки, содержащие данные о расстоянии определённого объекта от начала файла в байтах. Таким образом, программе, открывающей pdf файл, не обязательно загружать весь документ сразу, а можно открыть только представленную пользователю страницу.

На Рисунке 1 показаны структуры документов различных форматов.

Для лучшего понимания внутренней структуры docx и odt па рисунках 2,3 представлены фрагменты xml файлов.

Как видно из рисунков форматы docx и odt отличаются только большим количеством ссылок на другие объекты в docx, что может означать, что алгоритмы парсинга docx файлов могут подходить и для форматов odt с небольшими изменениями. При этом парсинг pdf документа будет проходить по другим правилам.

В odt и docx документах свойства и параметры текста хранятся при помощи тегов. Каждый определённый тег, примеры которых можно посмотреть в таблице 3, отвечает за одно определенное свойство параграфа или текста.

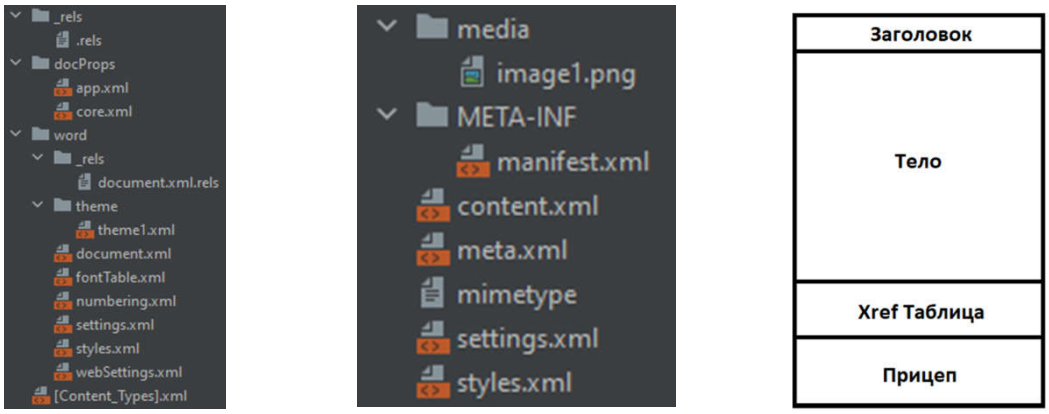

Рисунок 1 – Структура документов docx, odt и pdf соответственно

```
<office:text text:use-soft-page-breaks="true">
   <text:p text:style-name="P1">Показательный фрагмент</text:p>
 </office:text>
</office:body>
```

Рисунок 2 – Фрагмент xml файла odt

```
<w:body>
 <w:p w14:paraId="15F1A9B3" w14:textId="5E979413" w:rsidR="00DA18A4" w:rsidRDefault="002C0435">
  <w:r>
   <w:t>Показательный фрагмент</w:t>
  </w:r>
 </w:p>
```

Рисунок 3 – Фрагмент xml файла docx

Таблица 3

Основные используемые odt и docx теги

Значение	Тег docx	Тег odt
Отступ первой строки	w:firstline	fo:text-indent
Отступ слева	w:left	fo:margin-left
Интервал	w:line	fo:line-height
Кегель	w:sz	style:font-size
Шрифт	w:cs	style:font-name

PDF файл хранит информация о свойствах и параметрах в виде ключ-значения, используя специальные операторы. Их пример представлен в Таблице 4.

Таблица 4

Основные используемые в pdf операторы

Оператор	Значение
BT…ET	Начало и конец текста
Tf	Шрифт
Td, TD, Tm, T*	Позиционирование текста
Tj	Показать текстовую строку
TJ	Показать текстовую строку с учётом индивидуального позиционирования символов
Tl	Расстояние между строками
Tr	Опция рендеринга
Tc	Межсимвольное расстояние

Таким образом, docx и odt имеют похожую структуру и способ хранения свойств и параметров, а pdf представляется как совершенно отличный от других формат.

Особенности стилей в ODT. На Рисунке 4 представлена схема стилей для формата документов odt. Они состоят в иерархической структуре.

Главный родитель – стили по умолчанию**.** Весь их перечень представлен далее: chart, drawing-page, graphic, paragraph, presentation, ruby, table, table-cell, table-column, table-row, text. В разметке они находятся под тегом <style:default-style> в файле styles.xml. Все стили уровня ниже имеют ссылку на стиль по умолчанию, обозначаемый тегом <style:family>.

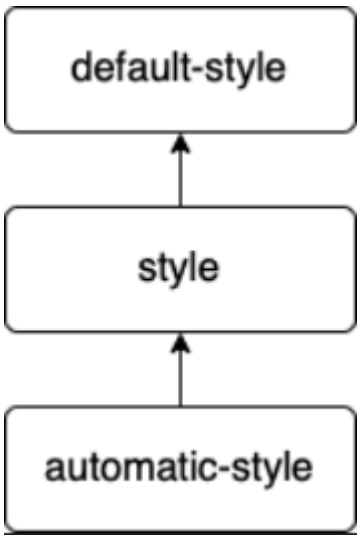

Рисунок 4 – Иерархия стилей

Следующий уровень стилей – общие стили. Они существуют под тегами <office:styles> и <style:style> в файле styles.xml. Общий стиль – это стиль, выбранный пользователем для документа или его части. Стили представляются пользователю как именованный шаблон свойств форматирования [13]. В большинстве редакторов их можно выбрать в панели быстрого доступа или в меню. Именно разработчики программного обеспечения определяют характеристики стиля и их наименование. Родителя стилей можно найти по атрибуту <style:parent-style-name>, а имя стиля – по <style:name>.

Нижний уровень схемы – автоматические стили. Они определяются элементами <office:automatic-styles> и <style:style> в файлах styles.xml и content.xml. В отличие от общих стилей они не представляются пользователю как таковые. Характеристики форматирования автоматического стиля предоставляются пользователю как свойства объекта, к которому он применяется [13]**.** Например, если автор написал определенный текст, используя общий стиль, но для конкретных слов изменил шрифт, от общего стиля наследуется автоматический, содержащий атрибут шрифта. Родителя и имя текущего стиля можно найти по тем же тэгам, что и у общих: <style:parent-style-name> и <style:name>.

Наследование стилей имеет еще ряд особенностей, которые нужно выделить. Автоматический стиль всегда наследуется только от общего. Однако общий может в родителях иметь дугой общий. И количество таких связей зависит исключительно от разработчиков текстового редактора. Если у общего нет родителя в своем уровне, то атрибут <style:parent-style-name> будет отсутствовать. Значит, следует обращаться к стилю по умолчанию. Однако при поиске параметров форматирования они могут не найтись и на верхнем уровне. Это значит, что требуется продолжить поиск в стилях объекта более высокого уровня. Например, в текст входит абзац, значит, если нужное не нашлось в семействе text, то требуется найти автоматический стиль текущего объекта из семейства paragraph. И далее путешествовать по нему. Дальше параграф может быть в ячейке таблицы, значит, параметр может быть в семействе table-cell [13].

Выбор технологий для реализации парсинга электронных документов. Для реализации парсинга электронных документов различных форматов были исследованы возможности языков Python и Java. Python является распространенным высокоуровневым языком программирования, и его популярность объяснима универсальностью и колоссальным числом различных библиотек. После анализа существующих модулей для работы с электронными документами форматов docx, pdf и odt были выбраны библиотеки: python-docx, pdfminer.six, odfpy. Java, как и Python, является распространенным высокоуровневым языком программирования, что обусловлено высокой скоростью исполнения и широкими возможностями по управлению памятью. По принципу наибольшего доверия от разработчиков для сравнения были выбраны библиотеки: Apache POI, PDFBox, Apache Tika.

Для того чтобы проверить, насколько широким функционалом обладают выбранные модули для языков Java и Python, были проведены эксперименты по извлечению различных данных из электронных документов формата docx, pdf и odt, результаты которых представлены в Таблицах 5–7.

Следует отметить, что популярностью при работе с документами формата odt на языке Java пользуются еще несколько библиотек. Например, JOpenDocument и ODFDOM. Было принято решение проводить исследование модуля Apache Tika, поскольку все другие библиотеки имеют одну из следующих проблем: они больше не поддерживаются или их добавление в проект неизбежно приводит к конфликту с другими модулями.

Полученные результаты. Сравнение форматов odt, docx, pdf показало, что процесс разработки функций, поддерживающих формат odt, будет проходить намного быстрее и легче по сравнению с pdf за счет схожей с docx документом структуры документов и способа хранения информации о контенте. В то же время поддержка формата pdf может вызвать некоторые трудности за счет

совершенно других принципов функционирования.

Сравнительный анализ возможностей языков программирования Java и Python показал, что при работе с электронными документами Python уступает Java только при извлечении данных из файлов формата pdf. Также благодаря широкому спектру использования языка программирования Python есть возможность применить его для создания серверной части сервиса автоматизированного нормоконтроля документов.

Таблица 5

Сравнение обработки docx-документа

Тест	Java		Python	
Извлечение текста из документа (18,6 тыс. символов)	0:00:01.525 мс	13 652 448 байт	**0:00:00.937 мс**	**9 598 128 байт**
Извлечение текста из таблиц документа	**0:00:01.112 мс**	10 494 504 байт	0:00:01.609 мс	**9 232 283 байт**
Выгрузка всех изображений из документа	0:00:01.071 мс	12 443 616 байт	**0:00:01.031 мс**	**9 599 132 байт**
Извлечение характеристик текста в документе	0:00:01.867 мс	9 647 072 байт	**0:00:01.562 мс**	**9 602 041 байт**
Значения отступов абзацев в документе	0:00:01.482 мс	**9 039 336 байт**	**0:00:00.968 мс**	9 599 256 байт

Таблица 6

Сравнение обработки pdf-документа

Тест	Java		Python	
Извлечение текста из документа объема 18,6 тыс. символов	**0:00:01.487 мс**	38 972 960 байт	0:00:13.156 мс	**37 531 819 байт**
Извлечение текста из документа объема 55,8 тыс. символов	**0:00:02.362 мс**	**12 569 680 байт**	0:00:23.328 мс	37 535 505 байт
Извлечение текста из таблиц документа	**0:00:01.060 мс**	**15 033 856 байт**	0:00:06.171 мс	37 478 435 байт
Извлечение характеристик текста в документе	**0:00:01.634 мс**	44 229 648 байт	0:00:13.671 мс	**37 558 904 байт**
Выгрузка всех изображений из документа	**0:00:01.299 мс**	**31 867 496 байт**	0:00:07.000 мс	37 491 904 байт

Таблица 7

Сравнение обработки odt-документа

Тест	Java		Python	
Извлечение текста из документа объема 18,6 тыс. символов	0:00:00.941 мс	**5 671 864 байт**	**0:00:00.640 мс**	18 745 631 байт
Извлечение текста из документа объема 37,2 тыс. символов	**0:00:00.839 мс**	**6 233 328 байт**	0:00:02.390 мс	22 937 261 байт
Извлечение текста из документа объема 55,8 тыс. символов	**0:00:00.984 мс**	**7 996 416 байт**	0:00:03.406 мс	28 595 347 байт

Выводы. Подводя итоги, можно сделать вывод, что парсинг электронных документов форматов docx, odt и pdf возможен. Однако при расширении функционала системы нормоконтроля документов, который на данном этапе имеет модуль парсинга docx, нужно отдать предпочтение odt формату.

Для парсинга документов оптимальным языком программирования определен Python, так как его модули обладают высокой скоростью и необходимым для проекта функционалом по сравнению с библиотеками Java.

Дальнейшие исследования предполагают следующие направления:

– разработка и реализация алгоритмов парсинга pdf, odt-файлов;

– оптимизация работы сервиса нормоконтроля документов;

– исследование возможности переноса сервиса на мобильную платформу.

Список источников

1. Гохберг Л.М. Образование в цифрах: 2021: краткий статистический сборник / Л.М. Гохберг, О.К. Озерова, Е.В. Саутина и др. // Национальный исследовательский институт «Высшая школа экономики». 2021. № 200. С. 64–51.

2. Отчет о научно-исследовательской работе. Структура и правила оформления. Введ. 2018-07-01. – М.: Стандартинформ. – 24 с.

3. Соловьёва Н.В., Гагарин А.В. Выпускная квалификационная работа как шаг к профессионализму, или 40 ошибок и 40 рекомендаций // Развитие профессионализма. 2018. № 2. С. 5–9.

4. Бережков А.В. Валитова Ю.О., Клименко А.И., Пономарев Д.Д. Опыт повышения качества оформления выпускных квалификационных работ студентов технического вуза // Педагогический журнал. Т. 10. 2020. № 1А. С. 367–375.

5. Свидетельство 2015615893. Нормоконтроль студенческих работ: программа для ЭВМ / И. А. Кашутина, О. О. Луковенкова, О. В. Кудринская (RU); правообладатель ФГБОУ ВО «КамГу им. Витуса Беринга»; заявл. 13.03.2015; опубл. 20.06.2015, Бюл. № 2015612356. 2,7 Мб.

6. Манкевич О.В., Семеняк П.А. Особенности автоматизации нормоконтроля текстовых документов // Информационные технологии и системы. 2015. С. 124–125.

References

1. Gokhberg L.M. Ozerova O.K., Sautina E.V. et al. Education in Numbers: 2021: a Brief Statistical Collection. *Nationalniy issledovatelskiy intstitut «Visshaya Shkola Economici»*. 2021. No. 200. pp. 64–51. (In Russ.).

2. Otchet o nauchno-issledovatelskoy rabote. Structura i pravila oformlenia. 2018-07-01. *Moscow. Standinform.* 2018. 24 p. (In Russ.).

3. Solovyova N.V., Gagarin A.V. Final Qualifying Work as a Step Towards Professionalism or 40 Mistakes and 40 Recommendations. *Razvitiye professionalizma.* 2018. No.2. pp. 5–9. (In Russ.).

4. Berezhkov A.V., Valitova Yu.O., Klimenko A.I., Ponomarev D.D. Experience of Improving the Quality of Registration of Final Qualifying Works of Students of a Technical University. *Pedagogicheskiy zurnal.* Vol. 10. 2020. No. 1A. pp. 367–375. (In Russ.).

5. Certificate 2015615893. Norm Control of Student Works: Computer Program / I.A. Kashutina, O.O. Lukovenkova, O.V. Kudrinskaya (RU); copyright holder *FGBOU VO «KamGU im. Vitusa Beringa»*; ap. 13.03.2015; publ. 20.06.2015, Bul. No. 2015612356. 2.7 Mb. (In Russ.).

6. Mankevich O.V., Semenyak P.A. Features of Automation of Standard Control of Text Documents *Informacionniye tekhnologii i sistemi.* 2015. pp. 124–125. (In Russ.).

7. Жигалова М.А., Сухов А.О. Автоматизированная система оценки соответствия текстовых документов требованиям // Коллоквиум молодых исследователей по программной инженерии. 2016. С. 135–140.
8. Стариченко Б.Е., Устинов М.А. Программа автоматизации контроля оформления текстовых документов // Педагогическое образование в России. 2018. №. 8. С. 163–168.
9. Поздин В.Н., Выймова Е.А. Алгоритм автоматизированного нормоконтроля работ обучающихся образовательного учреждения // Общество. Наука. Инновации (НПК-2021). № 2. 2021. С. 584–588.
10. Джонсон Д. Восемь самых популярных форматов документов в Интернете. 2014 [Электронный ресурс]. – Режим доступа: http://duff-johnson.com/2014/02/17/the-8-most-popular-document-formats-on-the-web /#data (In Eng.).
11. Джонсон Д. Популярность PDF в Интернете. 2021 [Электронный ресурс]. – Режим доступа: https://www.pdfa.org/pdfs-popularity-online/ (In Eng.).
12. Постановление Правительства Российской Федерации о создании Правительственной комиссии по импортозамещению и ее функций. 2015. № 785 [Электронный ресурс]. – Режим доступа: http://static.government.ru/media/files/gP7IKCc3BsBTtEQuYjUxArQ28Dr3oyA3.pdf
13. Формат открытого документа для офисных приложений (OpenDocument). 2010. № 1.2 (1) [Электронный ресурс] – Режим доступа: http://docs.oasis-open.org/office/v1.2/cd05/OpenDocument-v1.2-cd05-part1.html (In Eng.).

7. Zhigalova M.A. Automated System for Assessing Compliance of Text Documents with Requirements. *Kollokvium molodih issledovateley po programmnoy inzhenerii.* 2016. pp. 135–140. (In Russ.).

8. Starichenko B.E. Ustinov M.A. The Program of Automation of Control of Registration of Text Documents. *Pedagogicheskoye obrazovanie v Rossii.* 2018. No. 8. pp. 163–169. (In Russ.).
9. Pozdin V.N., Vymova E.A. Algorithm of Automated Norm Control of the Work of Students of an Educational Institution. *Obshchestvo. Nauka. Innovacii (NPK-2021).* 2021. No. 2. pp. 584–588. (In Russ.).
10. Johnson D. The Eight Most Popular Document Formats on the Internet. 2014. Available at: http://duff-johnson.com/2014/02/17/the-8-most-popular-document-formats-on-the-web/#data

11. Johnson D. The Popularity of PDF on the Internet. 2021. Available at: https://www.pdfa.org/pdfs-popularity-online/

12. *Postanovlenie Pravitelstva Rossiyskoy Federazii o sozdanii Pravitelstvennoy komissii po importozamesheniyu i ee funktsiy.* 2015. No. 785. Available at: http://static.government.ru/media/files/gP7IKCc3BsBTtEQuYjUxArQ28Dr3oyA3.pdf (In Russ.).
13. Open Document Format for Office Applications (OpenDocument). 2010. No. 1.2 (1). Available at: http://docs.oasis-open.org/office/v1.2/cd05/OpenDocument-v1.2-cd05-part1.html

Статья поступила в редакцию 30.06.2022; одобрена после рецензирования 12.07.2022; принята к публикации 14.09.2022.
The article was submitted 30.06.2022; approved after reviewing 12.07.2022; accepted for publication 14.09.2022.