УДК 004

ОПТИМИЗАЦИЯ АРХИТЕКТУРЫ СЕРВИСА АВТОМАТИЗИРОВАННОГО НОРМОКОНТРОЛЯ ВКР И ДИССЕРТАЦИЙ

Тартынских П.С. (магистрант, Университет ИТМО), **Лушников Д.М.** (магистрант, Университет ИТМО)

Научный руководитель – Насыров Н.Ф.

(ассистент, Университет ИТМО)

Аннотация.

В работе описывается использование различных подходов, которые в совокупности позволили существенно оптимизировать архитектуру онлайн-сервиса автоматизированного нормоконтроля. Рассмотрено изначальное состояние сервиса, обращено внимание на его недостатки и приведено обоснование необходимости оптимизации. Описано изменение бизнес-процессов проверки форматирования абзацев документа. Отдельно сделан акцент на использовании шаблонов стандартного оформления для предварительной классификации элементов docx документа. Описан процесс оптимизации серверной части системы. В докладе также приведены в пример метрики, которые демонстрируют успешность использования вышеперечисленных подходов.

Введение.

Гибкость архитектуры является одним из ключевых требований к информационной системе с целью ее сопровождения, адаптации под изменяющиеся требования бизнес-процессов и эффективного использования. Поводом для принятия решения об оптимизации архитектуры сервиса автоматизированного нормоконтроля стали новые функциональные требования, подводящие к необходимости автоматизации исправления ошибок оформленных элементов документов, а также требования к масштабируемости и отказоустойчивости системы. Потребовалось менять все слои архитектуры предприятия, были затронуты бизнес-процессы, информационный и технический слои, прикладные решения. В исследовании рассматриваются как конкретные практические аспекты реализации архитектуры, так и подходы по разделению логики на модули, описанные еще в 70-х Дэвидом Парнасом, концепции по описанию архитектуры, описанные Мэри Шоу и Дэвидом Гэрланом в 90-е.

Основная часть.

В основу разработки системы автоматизированной проверки соответствия требованиям оформления была положена клиент-серверная архитектура. Клиентский модуль представляет из себя приложение-надстройку для текстового редактора MS Word. Он взаимодействует с сервером, выполняющим всю работу по проверке оформления документа, посредством RESTful API. Одна из основных проблем анализа документов заключается в том, что при его создании одного визуального представления можно добиться несколькими способами. В связи с этим было решено разделить элементы документа на 49 классов, использовать для классификации параграфов в документах пользователей методы машинного обучения.

В изначальной версии сервиса его предполагалось использовать только для полной проверки всего docx документа и вывода на экран результатов проверки. Разработанный алгоритм подтвердил свою работоспособность, однако он имеет несколько особенностей, ограничивающих функционирование сервиса в условиях дальнейшего его развития:

- большая часть вычислений производится на стороне сервера,
- при каждой отправке документа на сервере проводится классификация его параграфов методами машинного обучения, что является избыточным в случае повторной проверки документа,
- при каждой новой классификации параграфов не учитываются замечания пользователя относительно корректности классификации,

- у пользователя отсутствует возможность проверить только необходимые ему части документа,

- у пользователя отсутствует возможность автоматически исправить элементы с ошибочным оформлением.

В связи с этим было решено оптимизировать архитектуру сервиса и разделить проверку всего документа на несколько функций проверки отдельных его частей, которые будут доступны пользователю. В качестве примера такой функции можно привести проверку форматирования абзацев документа. В общем виде весь процесс можно разделить на 4 стадии:

1) открытие документа в Word, загрузка надстройки, авторизация, выбор параметров проверки (отметка элементов, которые не подлежат проверке, выбор требований к оформлению и др.),

2) попытка определить классы проверяемых элементов с учетом анализа пресетов (стандартных шаблонов оформления и прошлых решений пользователя),

3) уточнение классов элементов пользователя и работа классификатора в случаях отсутствия однозначной интерпретации,

4) проверка форматирования отдельных абзацев и исправление/пропуск элементов других абзацев с аналогичным оформлением.

Также потребовались и изменения технической составляющей для реализации новых требований, появившихся в связи с изменением бизнес-процессов. Были определены несколько направлений, по которым осуществлялась оптимизация. Первым таким направлением стало изменение монолитного шаблона на микросервисный. Часть логики программы была вынесена в отдельный модуль, так как ранее при осуществлении классификации на основе обученной модели при каждой классификации избыточно загружались библиотеки машинного обучения. При проектировании микросервисной архитектуры использовался подход “API Gateway” для организации взаимодействия микросервисов. Это позволило повысить масштабируемость и отказоустойчивость.

Вторым направлением стало использование строго типизированного языка программирования Java вместо Python в большинстве программных модулей на стороне сервера.

В третьем направлении в ходе анализа технического и информационного слоев архитектуры сервиса было решено заменить первоначальную СУБД SQLite на PostgreSQL.

Выводы.

Выполненные в ходе данной работы изменения позволили привести сервис автоматизированного нормоконтроля к соответствию новым требованиям. Бизнес-процессы были дополнены новыми модулями, а количество запросов на сервер было уменьшено за счет использования системы пресетов и переноса части вычислительных операций по классификации абзацев на сторону клиента. Технический и информационный слои были трансформированы под отказоустойчивое многопользовательское приложение готовое к дальнейшему масштабированию.

Тартынских П.С. (автор) Подпись

Лушников Д.М. (автор) Подпись

Насыров Н.Ф. (научный руководитель) Подпись