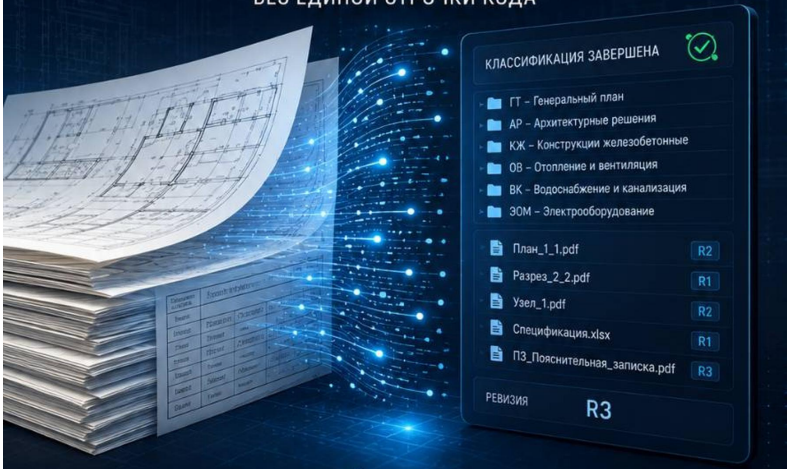
Андрей Леонов

CDC CORE

КАК НАУЧИТЬ КОМПЬЮТЕР РАЗБИРАТЬ СТРОИТЕЛЬНУЮ ДОКУМЕНТАЦИЮ

ИСТОРИЯ ПРОЕКТА — БОЛИ, РЕШЕНИЯ И ТЕХНОЛОГИИ
БЕЗ ЕДИНОЙ СТРОЧКИ КОДА



КОНВЕЙЕР ИЗ 9 ЭТАПОВ

1 СКАНИРОВАНИЕ И ЗАГРУЗКА
2 РАСПОЗНАВАНИЕ ТЕКСТА (OCR)
3 ИЗВЛЕЧЕНИЕ КОДОВ
4 НЕЧЕТКИЙ ПОИСК
5 ИЕРАРХИЯ ДОКУМЕНТА
6 РЕВИЗИИ И ВЕРСИОНИРОВАНИЕ
7 СТРУКТУРА ПАПОК
8 ПАСПОРТ И БАЗА ДАННЫХ
9 ОТЧЕТЫ И ПРОВЕРКА

АВТОМАТИЗАЦИЯ, КОТОРАЯ ЭКОНОМИТ ВРЕМЯ, ДЕНЬГИ И НЕРВЫ

Андрей Леонов

CDC Core. Как научить компьютер разбирать строительную документацию

https://litres.ru/74436654
SelfPub; 2026

Аннотация

Книга о том, как научить компьютер разбирать строительную проектную документацию. Тысячи чертежей, шифры комплектов, ревизии, иерархии разделов: ручная сортировка отнимает недели и плодит ошибки, которые оборачиваются переделками. CDC Core берёт рутину на себя: распознавание сканов, извлечение кодов, нечёткий поиск, раскладка по папкам и база назначений. Без единой строчки кода: история создания системы, конвейер из девяти этапов, технологии простыми словами, экономика внедрения, безопасность и чек-лист запуска за неделю. Исходный код открыт. Для руководителей, инженеров и всех, кто наводит порядок в документах.

Содержание

Андрей Леонов
CDC Core. Как научить компьютер разбирать строительную документацию

CDC Core. Как научить компьютер разбирать строительную документацию

История проекта, боли, решения и технологии — без единой строчки кода

Исходный код проекта: https://github.com/AndreyLeonov80/cdc-core
Telegram: https://t.me/aidialog
GitHub: https://github.com/AndreyLeonov80/about
E-mail: aidialog@mail.ru

Содержание

Введение. Зачем эта книга

Эта книга — история одного практического проекта и одновременно инструкция по мышлению. Ее герой — CDC Core, система автоматической классификации строительной проектной документации. Каждый день проектные организации выпускают сотни чертежей, пояснительных записок, спецификаций и смет. Каждый документ нужно распознать, понять, к какому комплекту он относится, определить его ревизию и положить в правильную папку. Вручную эта работа занимает часы и дни, а ошибка стоит дорого: документ, попавший не в тот комплект, может задержать стройку или привести к переделкам.

CDC Core берет эту рутину на себя. Система принимает сканы и текстовые выгрузки, распознает текст на русском языке, определяет код комплекта чертежей, выстраивает иерархию раздел — подраздел — часть — книга — том, фиксирует ревизию документа и раскладывает файлы по папкам. Результат сохраняется в базу данных и сводный отчет, с которыми работает человек-оператор через веб-интерфейс.

Книга написана без единой строчки исходного кода. Все технические решения объяснены словами: что делает каждый этап, зачем он нужен и какие задачи решает. Полный исходный код проекта открыт и доступен по ссылке: https://

github.com/AndreyLeonov80/cdc-core. Книга — это карта, код — территория. Читайте карту, а за деталями реализации заглядывайте в репозиторий.

Кому адресована книга? Руководителям проектных отделов, которые хотят понять, что реально может автоматизация. Инженерам, которые выбирают инструменты для обработки документации. Разработчикам, которым интересен разбор полного цикла: от сканированного листа до базы данных. И всем, кому любопытно, как устроены прикладные системы искусственного интеллекта изнутри — без математики и без кода, на уровне идей и решений.

Глава 1. Боль первая: хаос строительной документации

Строительный проект — это тысячи документов. Генеральный план, архитектурные решения, железобетонные конструкции, отопление и вентиляция, водоснабжение, электрика, технология производства, сметы, пояснительные записки. Каждый комплект чертежей имеет свой буквенный код: ГТ, АР, КЖ, ЭОМ, ОВ, ВК и десятки других. Внутри комплекта — разделы, подразделы, части, книги, тома. Отдельная ось — стадии строительства и ревизии: документ живет, меняется, и каждая его версия должна лежать на своем месте.

На практике документы приходят в самом разном виде. Сканы разного качества, фотографии с телефона, выгрузки из разных систем, файлы с именами вроде «скан_023_финал_2». Один и тот же чертеж может существовать в пяти копиях с разными именами в разных папках. Сотрудник открывает файл, вчитывается в штамп, определяет код комплекта, сверяется со структурой проекта, переименовывает файл и переносит его в нужную папку. На один документ уходят минуты. На тысячу — недели.

Цена ошибки высока. Чертеж, попавший в чужой комплект, не найдут, когда он понадобится. Устаревшая реви-

зия, выданная в работу вместо новой, превращается в переделки на площадке. Потерянный документ приходится запрашивать заново. Хаос в документации — это не абстрактное неудобство, а прямые финансовые потери и срывы сроков.

Первая боль, которую решает CDC Core, — это именно хаос на входе. Система не требует, чтобы файлы были аккуратно названы или разложены. Она принимает то, что есть: гору сканов и текстовых файлов — и сама разбирает, что есть что. Вместо недель ручной сортировки — автоматический прогон, после которого человек проверяет результат, а не выполняет его с нуля.

Глава 2. Боль вторая: ручная работа с кодами и ревизиями

Даже когда документы более-менее упорядочены, остается тонкая ручная работа: определить код комплекта и ревизию. Код комплекта — это короткий шифр из букв и цифр, который нужно найти в тексте документа. Он может стоять в штампе, в колонтитуле, в названии файла или прятаться среди десятков других обозначений. Человек учится видеть его с опытом, но каждый новый проект приносит новые форматы.

Ревизия — отдельная головная боль. Документ изменяется: вышло изменение номер один, потом номер два. Каждая ревизия должна храниться отдельно, и всегда должно быть понятно, какая версия актуальна. В ручном режиме ревизии путаются: файл перезаписывают поверх старого, номер изменения забывают обновить, актуальную версию ищут по дате файла, которая ничего не гарантирует.

Есть и ловушки, в которые попадает даже внимательный человек. Дата в тексте похожа на код документа: набор цифр с точками и дефисами. Номер страницы похож на ревизию. Служебные пометки — «Раздел», «Книга», «Часть» — сбивают с толку при поиске. Правила, отличающие настоящий код от похожего мусора, приходится держать в голове и при-

менять к каждой строке.

Вторая боль, которую снимает CDC Core, — это автоматическое извлечение кодов и ревизий по формальным правилам. Система проверяет каждую строку текста набором эвристик: достаточно ли в ней цифр, нет ли признаков даты, нет ли служебных слов, есть ли характерные символы. Кандидаты, прошедшие фильтр, сравниваются со справочником нечетким поиском, который прощает опечатки распознавания. Человеку остается контрольная функция: подтвердить или поправить, а не выискивать.

Глава 3. Боль третья: система, которую нельзя развивать

Третья боль менее заметна, но со временем становится главной. Первый рабочий прототип системы обработки документов обычно рождается как один большой файл: все подряд — веб-сервер, база данных, распознавание, поиск, отчеты. Пока задач мало, это удобно: все под рукой, все работает. Но проект растет. Добавляется новый тип документов — приходится лезть в середину файла и бояться что-нибудь сломать. Нужно изменить одну эвристику — непонятно, на что еще она влияет. Новый разработчик открывает файл на несколько тысяч строк и тонет.

Монолит тянет за собой и другие проблемы. Настройки захардкожены: адреса, пути к папкам, порты разбросаны по коду. Пароли и ключи лежат рядом с логикой. Запуск зависит от конкретной машины разработчика: на другом компьютере система не стартует, потому что ждет папку, которая есть только на первом. Тестов нет, потому что тестировать клубок взаимосвязанного кода почти невозможно. Каждое изменение — риск, каждый релиз — стресс.

Рано или поздно наступает момент, когда развивать систему дороже, чем переписать ее. Именно этот путь прошел CDC Core: первая версия была монолитом в одном файле на

несколько тысяч строк, а зрелая версия — аккуратным пакетом из небольших модулей, где у каждого своя ответственность. Переход не был cosmetic: изменились имена, структура, способ хранения настроек, подход к ошибкам. Зато результатом стала система, которую можно тестировать, документировать и развивать по частям.

Третья боль, которую закрывает проект, — это боль роста. CDC Core спроектирован так, чтобы новый тип документов, новый алгоритм сравнения или новый отчет добавлялись как отдельный модуль, а не как правка в середине гигантского файла. Эта книга в том числе и об этом: как принимать архитектурные решения, которые не мешают будущему.

Глава 4. Как создавался CDC Core

История проекта началась с простой задачи: перестать разбирать сканы вручную. Первая версия умела немногое — принять изображение, распознать русский текст и сохранить результат. Она работала, и именно работающий прототип стал фундаментом: на нем проверялись идеи, которые потом вошли в зрелую систему.

Второй этап — извлечение смысла. Распознанный текст сам по себе бесполезен, если из него нельзя достать код комплекта, ревизию и стадию. Появились правила поиска кодов, справочники комплектов и объектов, нечеткое сравнение для прощения ошибок распознавания. Система научи-

лась отличать настоящий код от даты, номера страницы и служебных пометок.

Третий этап — иерархия. Документу мало иметь код: нужно понять его место в структуре проекта. Появились разбор дерева разделов, поиск по иерархическому справочнику и sidecar-файлы — маленькие паспорта, которые хранятся рядом с каждым текстовым файлом и помнят все, что система о нем узнала: ревизию, стадию, объект, коды разделов.

Конец ознакомительного фрагмента.