22 октября 2025

No-code без мифов: 10 выводов из реальных проектов

За 20 лет в ИТ мы видели, как много проектов по внедрению СЭД и ECM сталкиваются с трудностями не из-за плохого кода, а из-за бесконечных изменений требований, разрыва между бизнесом и ИТ и запредельной стоимости поддержки. Мы создали платформу «Авандок» и ее no-code конструктор как ответ на эти вызовы.

«Авандок.Платформа» — решение для комплексного управления корпоративной информацией: электронными документами, контентом и связанными с ними процессами.

Наша целевая аудитория — средний и крупный бизнес, обрабатывающий большие объемы документов и имеющий уникальные, сложные процессы в области документооборота, архивного делопроизводства, управления контентом и другой корпоративной информацией (заявки, управление услугами, активами и другими процессами). Вместе с тем, «Авандок» может быть оптимальным решением и для малого бизнеса, где тиражируемые продукты не способны удовлетворить специфические требования заказчика.

Ключевой элемент «Авандок» — мощный no-code конструктор, который мы развиваем уже более 5 лет. Он дает возможность решать самые сложные задачи без привлечения разработчиков. При этом платформа обеспечивает

производительность, масштабируемость и отказоустойчивость необходимые в enterprise-среде. За это время мы внедрили его в компаниях разного масштаба, накопили значительный практический опыт и развеяли многие мифы об этом подходе.

Ниже — выводы, к которым мы пришли в ходе реализации крупных внедрений и общения с заказчиками.

Мифы и реальность no-code

Вывод №1. No-code не позволяет создавать приложения enterprise уровня специалистами без технической подготовки

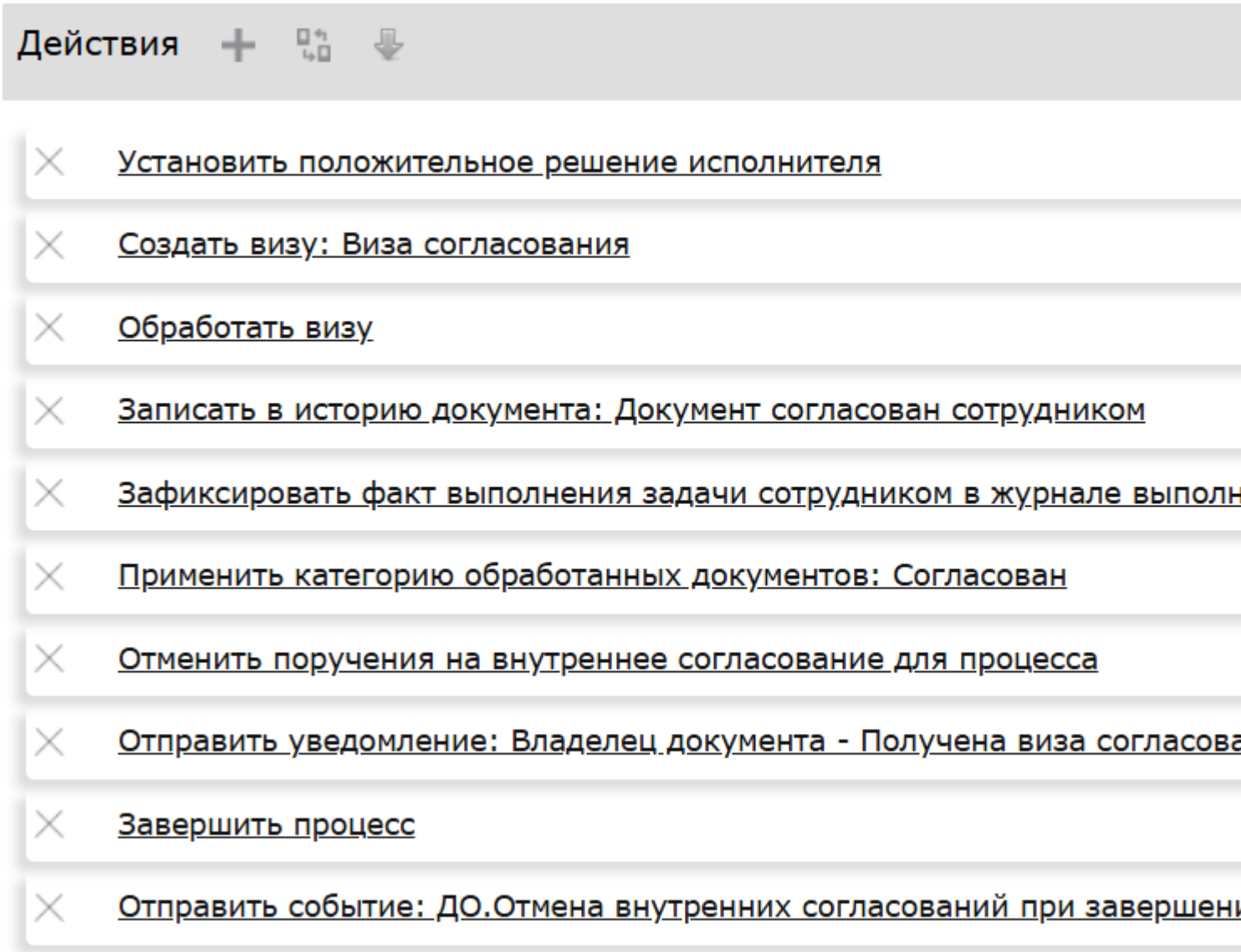
Создание решений на no-code-инструментах требует системного мышления и глубокого понимания логики построения процессов. Это по-прежнему разработка — только на другом уровне абстракции. Эффективно работать в конструкторе можно лишь при наличии технического бэкграунда.

Приведенные ниже аспекты охватывают лишь часть того, что требует внимания при работе с no-code:

- Объектная модель процесса: набор и свойства мета данных процесса
- Работа со справочниками: настройка объектной модели, экранных форм, табличных представлений, правил фильтрации и т.д.
- Наследование: для любого компонента (кнопка, закладка, поле, разметка и др.) необходимо иметь возможность настройки наследования, позволяющей определить различные варианты поведения этого компонента в зависимости от условий.
- Контекст: условия применения настроек для компонента в зависимости от: роли, владельца объекта, значения заданного атрибута, прав доступа, состояния текущего процесса и т.д.

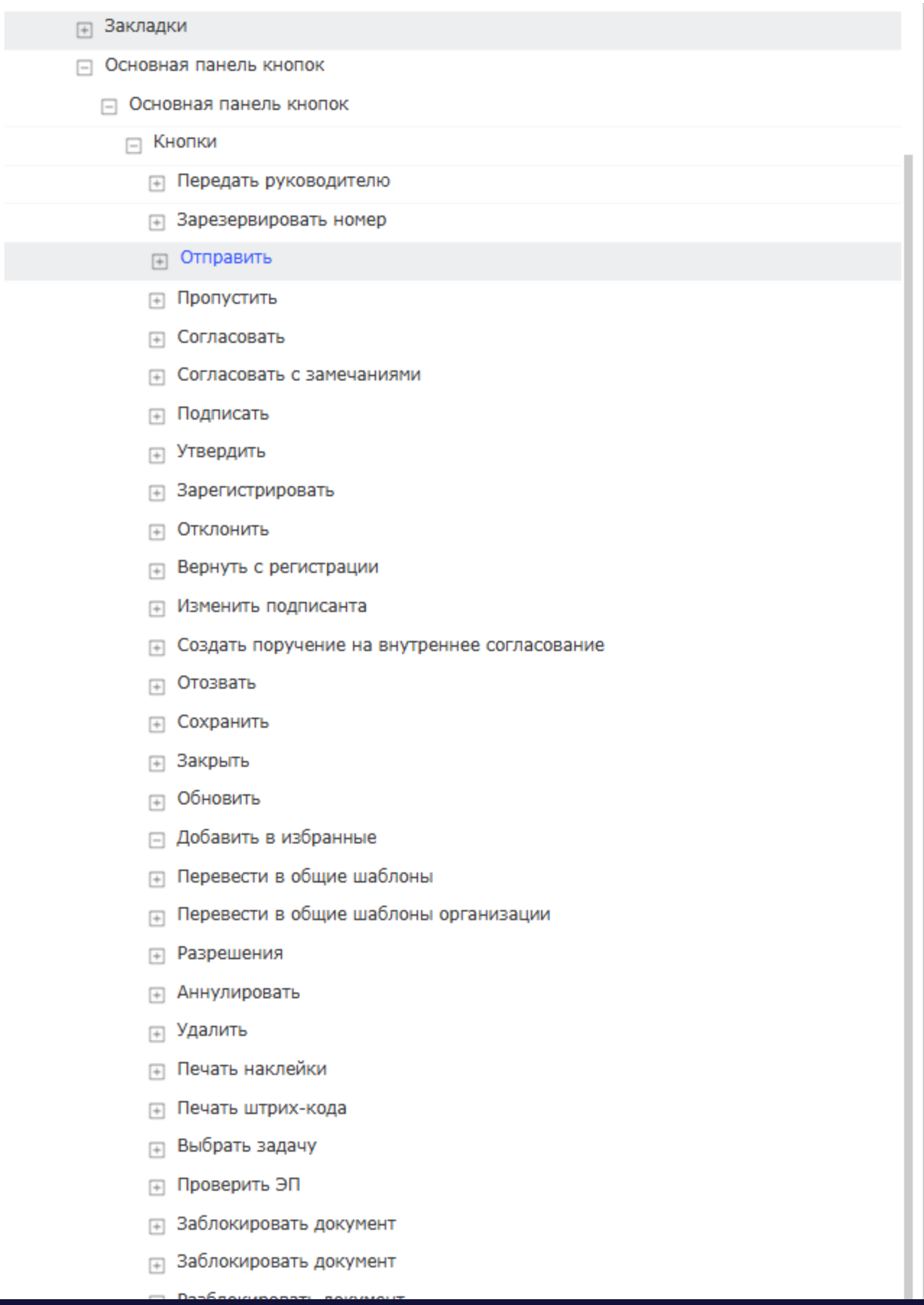
- Атрибуты: состав и группировка атрибутов, размещение и правила отображения атрибутов в зависимости от различных условий; стили метки или значения атрибута, правила валидации, текст помощи, условия поиска и т.д.
- Процесс: описание состояний, обработчиков событий и действий.

Ниже пример набора действий, выполняемых при согласовании:



Панель кнопок: для каждой кнопки необходимо определить большой перечень валидаторов, контекстов, условий наследования и выполняемых

атомарных действий. Например, для одного только организационного документа может потребоваться сложная конфигурация набора кнопок — и это будет индивидуально для каждого крупного заказчика. Отличаться может не только состав, но и логика отображения и исполнения каждой кнопки.


Закладки
Основная панель кнопок
Основная панель кнопок
Кнопки
Передать руководителю
Зарезервировать номер
Отправить
Пропустить
Согласовать
Согласовать с замечаниями
Подписать
Утвердить
Зарегистрировать
Отклонить
Вернуть с регистрации
Изменить подписанта
Создать поручение на внутреннее согласование
Отозвать
Сохранить
Закрыть
Обновить
Добавить в избранные
Перевести в общие шаблоны
Перевести в общие шаблоны организации
Разрешения
Аннулировать
Удалить
Печать наклейки
Печать штрих-кода
Выбрать задачу
Проверить ЭП
Заблокировать документ
Заблокировать документ
Сформирова
Наследова
Дейст
Дейст
Дейст
Дейст
Дейст
Дейст
Дейст
Контекст
Наименован
Отправить
Разместить
Недоступно
Скрыть в пр
Валидатор
1. Гру
2. Гру
Групп
3. Гру
Действия
Закры

Кроме этого, необходимо настроить:

- представление процесса в списках (состав и порядок колонок, их ширина по умолчанию),
- печатные формы (по шаблонам и условиям),
- систему уведомлений (условия отправки, тексты и получатели),
- а также десятки других элементов.

Конструктор платформы «Авандок» включает:

- 37 уникальных инструментов
- 400+ компонентов
- 1000+ настроек

Вывод №2. No-code отлично подходит для создания MVP и небольших решений

Прототипирование, проверка гипотез и первичные запуски занимают меньше времени. Это позволяет бизнес-заказчикам буквально за дни, а не месяцы, увидеть работающий прототип будущего процесса, «пощупать» его и внести правки до начала масштабной и дорогостоящей разработки. Ключевое преимущество здесь — возможность проводить итерации с непосредственным участием бизнес-пользователей и мгновенно воплощать их обратную связь в обновленной версии прототипа. Это не просто ускоряет старт проекта, но и дает заказчику чувство контроля на самом раннем этапе. Фактически, no-code позволяет сместить фокус с обсуждения абстрактных требований на бумаге к совместной работе над живым, работающим решением.

No-code инструменты платформы «Авандок» позволяют создать прототип рабочего процесса всего за пару часов. А полный MVP — с экранной формой, бизнес-процессом, дашбордом с ключевой информацией, отчетом по процессу и готовый к демонстрации заказчику — можно собрать за 1 день.

Вывод №3. No-code не заменяет, а дополняет традиционное программирование

Любой no-code инструмент ограничен возможностями, заранее предусмотренными в платформе. Все, что выходит за рамки конструктора, потребует вмешательства разработчиков. Однако именно в этой связке и кроется главная сила подхода. No-code идеален для быстрой сборки типовых бизнес-сценариев и MVP: согласований, заявок, карточек данных, маршрутизации задач, а также при проектировании новых бизнес-сценариев. Это слой прикладной бизнес-логики.

Традиционное программирование отвечает за создание низкоуровневых сервисов, сложных интеграций, алгоритмов машинного обучения, и конечно же создания новых инструментов конструктора — то есть за расширение самих возможностей платформы.

Таким образом, no-code становится новой возможностью, позволяя разработчикам не тратить время на рутинный код, а сосредоточиться на создании уникальной ценности и сложной функциональности, передав сборку типовых сценариев бизнес-архитекторам и аналитикам.

В наших крупных проектах доля участия разработчиков сократилась с 30% до 10%, т.е. более 80% задач, связанных с адаптацией платформы под требования заказчика, выполняются с помощью конструктора, а в небольших проектах до 100% задач.

Стратегические выгоды для бизнеса

Вывод №4. Заказчику важны не технологии (no-code), а бизнес-преимущества, которые они дают в долгосрочной перспективе

На этапе внедрения заказчик фокусируется на сроках, бюджете и соответствии ТЗ. Прямая ценность no-code может быть неочевидна. Однако это ключевое инвестиционное преимущество, которое страхует от главных рисков будущего:

- Риск изменения требований: Бизнес-процессы не статичны. No-code позволяет адаптировать систему к новым требованиям с минимизацией затрат.
- Риск зависимости от вендора: Классическая разработка закрепляет зависимость от команды разработчиков вендора. No-code передает контроль над логикой заказчику: он может самостоятельно вносить изменения, уменьшая затраты на сопровождение. В большинстве случаев не требуется участие Senior-разработчиков — конструктора достаточно.
- Риск устаревания: Скорость изменений в бизнесе опережает возможности классической разработки. No-code выступает как инструмент цифровой трансформации, позволяя бизнесу оставаться гибким.

Таким образом, no-code из технологической фичи становится стратегическим инструментом для заказчика, обеспечивающим гибкость, независимость и управляемость.

Вывод №5. No-code меняет не скорость, а сам процесс внедрения, повышая его предсказуемость

Утверждение, что no-code магически сокращает сроки проекта — миф. Скорость внедрения по-прежнему определяется тремя факторами: качеством технического задания, скоростью согласований на стороне заказчика и соблюдением проектной дисциплины. Однако no-code значительно меняет внутреннюю структуру проекта. Он существенно сокращает цикл обратной связи: прототип превращается в работающую функциональность за дни или даже часы, а доработки вносятся «на лету». Это не отменяет этапы сбора требований, тестирования и документирования, но делает сам процесс разработки гораздо более гибким и отзывчивым к изменениям.

Одним из ключевых преимуществ становится стандартизация и повторное использование компонентов. Во многих решениях повторяются типовые элементы: наборы обработчиков кнопок, сценарии действий, шаблоны процессов.

«Авандок» позволяет создавать библиотеки таких компонентов, которые можно копировать и переиспользовать в разных решениях. Это не только ускоряет разработку, но и обеспечивает единообразие, снижает количество ошибок и упрощает поддержку.

Вывод №6. Центр компетенций: следующий шаг

Наличие no-code-инструментов меняет модель работы с вендором. Крупные внедрения показывают, что зрелый заказчик приходит к необходимости создания внутреннего Центра компетенций (ЦК) — команды специалистов, которые глубоко знают бизнес-процессы компании и возможности платформы.

Такой ЦК становится ключевым преимуществом организации:

- Ускоряет реализацию идей: изменения и доработки вносятся силами внутренней команды, без долгого ожидания внешних исполнителей.
- Снижает затраты и риски: сокращается зависимость от подрядчиков.
- Повышает качество решений: команда, погруженная в контекст бизнеса, создает более точные и эффективные конфигурации.

Таким образом, no-code — это катализатор изменений, который позволяет организации самостоятельно управлять своей ИТ-средой, делая ее гибкой и соответствующей уникальным бизнес-процессам. Формирование ЦК — это шаг от роли зависимого заказчика к роли уверенного оператора собственной цифровой экосистемы.

Как мы помогаем создать Центр компетенций:

- Совместное участие в пилотном процессе наших и ваших специалистов по завершении основного этапа внедрения.
- Обучение вашей команды по работе с конструктором.
- Консультативная поддержка плюс помощь с аудитом решений, реализованных в конструкторе.

Инженерная зрелость платформы

Вывод №7. No-code позволяет создавать производительные и масштабируемые решения

Мощная платформа обеспечивает базовую производительность и масштабируемость, но ключевым фактором остается качество проектирования бизнес-логики. No-code дает свободу конфигурирования, но требует архитектурной грамотности: платформа гарантирует надежность среды исполнения, а архитектор процессов отвечает за оптимальность конфигураций.

Для поддержания высокой производительности критически важны как ручной контроль с нагрузочным тестированием, так и инструменты мониторинга и анализа выполнения операций. Это позволяет выявлять и устранять проблемы до выхода в продуктив. Наличие встроенных инструментов мониторинга является конкурентным преимуществом и существенно снижает операционные риски для enterprise-заказчиков.

Производительность платформы «Авандок» многократно подтверждалась нагрузочными тестами, которые проводили наши клиенты перед запуском системы. В некоторых случаях такие тесты выполнялись в рамках конкурсных процедур, то есть еще до выбора продукта. На сегодняшний день мы можем смело утверждать, что «Авандок» одна из самых производительных платформ на рынке:

- 70 000 одновременных пользователей
- 1 000 000 операций в час
- 200 документов в секунду

Вывод №8. Инструменты отладки и контроля версий: основа командной работы

Когда над проектом одновременно работают несколько специалистов (в крупных проектах — десятки), no-code-конструктор перестает быть просто

инструментом индивидуальной разработки и становится средой для коллективной работы. В таких условиях критически важны встроенные механизмы, обеспечивающие предсказуемость и контролируемость процесса разработки.

Ключевыми становятся два типа инструментов:

- Контроль версий: возможность отслеживать историю изменений, сравнивать разные версии настроек, фиксировать авторство правок и при необходимости откатываться к предыдущим стабильным состояниям. Это защищает от хаотичных изменений и конфликтов между разработчиками.
- Инструменты отладки: возможность пошагово выполнять бизнес-процессы, отслеживать состояние данных на каждом этапе, визуализировать логику принятия решений и моментально находить источник ошибки. Это кардинально сокращает время на поиск и устранение проблем.

Таким образом, эти инструменты — не просто «удобство», а обязательное требование для промышленной эксплуатации no-code в проектах уровня enterprise. Они превращают разработку в управляемый инженерный процесс, обеспечивают надежность, повторяемость и высокое качество создаваемых бизнес-решений. Без них масштабная командная работа над сложными процессами становится невозможной.

В «Авандок» реализованы инструменты для работы с конструктором, которые позволяют пользователю, настраивающему решения, сравнить версии одного или нескольких компонентов, просматривать выполненные изменения до сохранения и проверять, какие действия, события, запросы к СУБД выполняются в рамках исполнения экземпляра бизнес-процесса.

Вывод №9. Документация: от рутины к автоматической актуальности

Гибкость no-code, позволяющая вносить изменения «на лету», создает классическую проблему enterprise-разработки: документация устаревает быстрее, чем создается. Ручное ведение документации в такой среде неэффективно и почти всегда бесполезно.

Единственное решение — автоматическая генерация документации непосредственно из текущих настроек платформы. Встроенные инструменты должны уметь создавать актуальные срезы системы: визуализировать схемы процессов, отражать все условия ветвления, фиксировать привязанные обработчики и правила. Это превращает документацию из формального отчета в живой, всегда доступный и достоверный источник информации.

Таким образом, автоматическая документация становится частью no-code-платформы, обеспечивая прозрачность, упрощая аудит и анализ сложных процессов, а также значительно снижая порог входа для новых членов команды. Это обязательный элемент зрелой платформы, который закрывает ключевые операционные риски enterprise-заказчика.

Эволюция платформы

Вывод №10. Развитие no-code платформы — это постоянный процесс

Развитие no-code платформы — это параллельный процесс по четырем направлениям:

1. Эволюция конструктора

Нельзя «однажды разработать» и забыть. Конструктор должен постоянно получать новые, более сложные и гибкие инструменты (например, работа с сложными объектными моделями, расширенные условия, новые визуальные

элементы), чтобы бизнес-архитекторы могли воплощать любые новые идеи без перехода на код.

2. Расширение базовой функциональности платформы

Функциональность конструктора бессмысленна без мощного фундамента. Мы непрерывно развиваем ядро: интеграционные модули, AI-сервисы (анализ текста, классификация), криптография, поддержка современных СУБД и общесистемного программного обеспечения. Это гарантирует, что собранные в конструкторе процессы будут работать быстро, надежно и безопасно в любой enterprise-среде.

3. Информационная безопасность

Для enterprise безопасность — не фича, а базовое требование. Наш процесс разработки включает не только регулярное обновление библиотек и мониторинг уязвимостей (CVE), но и встроенные в конструктор инструменты для соблюдения внутренних политик безопасности заказчика (например, настройка аудита и правил доступа без участия программистов).

4. Разработка новых функциональных модулей на инструментах конструктора

Важно не забывать, что заказчика, в первую очередь интересует не сама платформа как продукт, а то какие задачи с помощью этой платформы можно решать.

Поэтому ключевое направление развития — использование no-code конструктора для быстрой сборки и поставки готовых функциональных модулей (например, «Договорной документооборот», «Кадровое делопроизводство», «Сервис-деск»). Эти модули не являются жесткими «коробками» — заказчик может гибко адаптировать их под свои процессы все тем же конструктором.

Это создает двойную выгоду:

- Для заказчика: это кратно повышает ценность платформы, так как он получает не просто инструмент, а готовые решения для конкретных задач, что сокращает сроки и стоимость внедрения.
- Для команды разработки/внедрения: мы предлагаем не просто инструмент, а готовые решения для конкретных задач, что кратно повышает ценность платформы и ее конкурентоспособность.

Что в итоге

Главный итог наших пяти лет работы: no-code не заменяет профессиональную разработку, а меняет ее структуру. Это не инструмент для неподготовленных специалистов, а способ разделения труда: разработчики фокусируются на сложных технических задачах (интеграции, алгоритмы, ядро платформы), а бизнес-аналитики и бизнес-архитекторы — на настройке бизнес-логики.

Успех приходит, когда команда работает по принципу «каждый на своем месте»: технические специалисты создают надежный фундамент, а бизнес-специалисты настраивают процессы. No-code не отменяет требований к дисциплине, архитектуре и контролю качества — просто эти задачи теперь решаются на другом уровне. При этом системное мышление и контроль версий помогают проекту оставаться стабильным и управляемым.

В результате бизнес быстрее получает необходимые функции, а ИТ-команда может фокусироваться на действительно сложных и интересных задачах. Это практичный подход, который делает процесс создания ПО более эффективным и ориентированным на бизнес-результаты.