Бесплатный гид

1С и Telegram-боты: три практических разбора

Схема обмена и грабли · Сколько стоит доработка 1С · Как провести приёмку

Три разбора на практике: как устроена связка 1С ↔ Telegram-бот, как честно посчитать стоимость доработки и что проверять на приёмке, чтобы не переделывать потом. Без воды, с формулами, чек-листами и рабочими примерами.

Внутри	Формат
Три части подряд, единая нумерация разделов	PDF, A4, шрифты встроены
Таблицы, списки и примеры расчёта	Пример: выгрузка остатков в Telegram-бот
Чек-лист приёмки из 7 блоков	Блокировки, откат, безопасность, эксплуатация
Блок «Что дальше» в конце	Канал @onec_tg и витрина продуктов

Автор: Александр Ким · 1С-разработка и Telegram-боты

Текст подготовлен с помощью ИИ-агента и вычитан автором.

Оглавление

Гайд состоит из трёх частей. Каждая часть — самостоятельный разбор; читать можно по порядку или по нужной теме.

Часть 1. Как я связал 1С с Telegram-ботом: схема обмена и грабли

- Почему сначала выбирают схему, а не пишут код
- Схема 1: Telegram → бот → HTTP-сервис 1С
- Грабли: long polling, кодировка, JSON, один сервис на всё
- Схема 2: 1С → Telegram (уведомления)
- Что я делаю на практике
- Безопасность: токены, HTTPS, secret token

Часть 2. Сколько стоит доработка 1С: почему «от 30 000 ₽» не помогает

- Почему «от 30 000 ₽» — не оценка
- Как считаю я: шесть шагов методики
- Пример: выгрузка остатков в Telegram-бот
- Что чаще всего забывают в оценке

Часть 3. Приёмка 1С-разработчика: что проверять, чтобы не переделывать

- Функциональность — по сценарию, а не по списку
- Данные
- Интеграции
- Производительность
- Безопасность и права
- Эксплуатация
- Сводка и замечания

Что дальше

Канал с разборами и шаблоны документов — в конце гайда.

Часть 1

Как я связал 1С с Telegram-ботом: схема обмена и грабли

Когда клиент просит «сделать бота, который отвечает в 1С», почти всегда выясняется, что непонятно одно: кто инициирует обмен. Бот пишет человеку, человек жмёт кнопку — и дальше 1С должна что-то сделать. Или наоборот: в 1С изменился документ, и бот должен написать клиенту. Обе схемы живут, но ломаются по-разному, и я обычно начинаю с выбора схемы, а не с кода.

Почему сначала выбирают схему, а не пишут код

Ответ на этот вопрос задаёт всё остальное: где будет жить логика, какие права нужны учётной записи, где появятся таймауты и в каком месте потом искать причину сбоя. Ниже — две рабочие схемы и грабли, которые встречаются в каждой из них.

Схема 1: Telegram → бот → HTTP-сервис 1С

Это основная рабочая схема. Бот получает апдейт от Telegram, разбирает команду, дёргает HTTP-сервис 1С и отвечает пользователю.

Что важно в самой 1С:

• **HTTP-сервис живёт на платформе, а не в конфигурации.** Конфигурацию трогать не нужно, если задача узкая.

• **Данные передаются как JSON или XML.** Я почти всегда выбираю JSON: меньше шума в разборе и проще отлаживать.

• **Сервис должен отвечать быстро.** Если обработка длинная (закрытие месяца, пересчёт партий), HTTP-сервис — плохое место: пользователь увидит таймаут, а не результат.

• **Права и пароли.** Сервис живёт под своей учёткой, у неё ровно те права, которые нужны на чтение или запись нужных объектов.

Грабли, на которые я натыкался:

1. Long polling и таймауты. Telegram-бот на polling'е держит соединение 30–60 секунд. Если 1С отвечает дольше — клиент получает «повторите запрос», и пользователь жмёт кнопку ещё раз. Двойной запрос, задвоенная запись.

2. Кодировка. Ответ 1С в UTF-8, а клиент ждёт CP1251 — классика. Я всегда явно задаю кодировку в заголовке ответа.

3. JSON в 1С без строковых обходов. Числа приходят как числа, даты — как строки, булевы — как **true/false**. Наивный разбор ломается на **null** в необязательных реквизитах.

4. Один веб-сервис на всё. Схема ломается, когда через один сервис гоняют и «покажи заказ», и «запиши оплату»: разные права, разные таймауты, разные клиенты.

Схема 2: 1С → Telegram (уведомления)

Здесь инициатор — сама 1С. В конфигурации ставят регламентное задание, которое смотрит нужные изменения и вызывает HTTP-сервис Telegram (sendMessage) или подключает внешнюю компоненту.

Что тут неприятно:

• Регламентные задания на сервере 1С живут своей жизнью: их можно остановить, они могут не стартовать после рестарта сервера, у них своё расписание, которое кто-то переставил.

• Изменения «в базе» и «у пользователя» — разные вещи. Уведомление из регламентного задания придёт, даже если пользователь, сделавший документ, его потом отменил.

• Нужен журнал отправок. Без него «бот не отвечает» и «событие не наступило» неразличимы.

Сравнение схем

Критерий	Telegram → бот → HTTP-сервис 1С	1С → Telegram (уведомления)
Кто инициирует	Пользователь в чате	Регламентное задание в 1С
Где живёт логика	В 1С, бот — тонкий слой	В конфигурации 1С
Главный риск	Таймауты и двойные запросы	Событие наступило, но не актуально

Критерий	Telegram → бот → HTTP-сервис 1С	1С → Telegram (уведомления)
Обязательный элемент	Учётная запись с узкими правами	Журнал отправок
Где искать причину сбоя	Ответ сервиса и лог бота	Журнал отправок и расписание задания

Свой опыт: дольше всего я возился именно с уведомлениями — с журналом и с тем, чтобы при отмене документа уведомление не уходило.

Что я делаю на практике

Для учётных сценариев — первая схема. Бот — тонкий слой: принимает команду, валидирует, дёргает сервис, отвечает. Логика живёт в 1С.

Для уведомлений — вторую, но с обязательным журналом отправок и с проверкой «актуально ли ещё событие» перед отправкой.

Отдельно про безопасность: токен бота не должен лежать в коде. Обычно он хранится в настройках базы или в переменной окружения на сервере, а обмен с 1С идёт по HTTPS с проверкой источника. Подпись запроса от Telegram (secret token) — полезная вещь, и её стоит включать сразу, потому что потом «просто добавить» уже поздно.

Безопасность: короткий чек-лист

Что	Как правильно
Токен бота	В настройках базы или переменная окружения, не в коде
Канал связи	HTTPS с проверкой источника запроса
Подпись Telegram	secret token включить сразу, на этапе внедрения
Учётная запись сервиса	Ровно те права, которые нужны на нужные объекты

Часть 2

Сколько стоит доработка 1С: почему «от 30 000 ₽» не помогает

Почти каждое ТЗ на доработку 1С начинается со слов «нужно посмотреть и оценить». И почти каждое заканчивается цифрой, которую никто не может объяснить: ни заказчик, ни исполнитель. Я вижу это постоянно, и за годы у меня сложился способ считать, который хотя бы можно разобрать по частям.

Почему «от 30 000 ₽» — не оценка

Эта цифра почти никогда не связана с объёмом работы. Обычно она означает одно из трёх:

- «столько стоит самый дешёвый кусок, чтобы было неприлично мало»;
- «так было в прошлом проекте, где задача была похожая, но не такая»;
- «назовём число, чтобы не торговаться».

Проблема в том, что между «похожей» и «такой же» в 1С пролегает качественная разница. Доработка отчёта и доработка, которая пересчитывает остатки по всем регистрам, — это разные объёмы, хотя обе звучат как «доработка отчёта».

Как считаю я: шесть шагов методики

1. Ставка человеко-дня.

Считаю от ставки, а не от «сложности». У меня базовая ставка человеко-дня, от которой я отталкиваюсь в расчётах. Дальше всё умножается на коэффициенты.

2. Разбивка на блоки.

Задача делится на блоки, и каждый оценивается отдельно: анализ и обследование, изменение конфигурации, обмен данными, тестирование, документация, внедрение. Оценка «целиком» всегда врёт, потому что блоки оцениваются по-разному.

3. Коэффициент сложности на блок.

Обычные 1.0. Доработка, затрагивающая несколько регистров и требующая проверки на существующих данных, — 1.3–1.5. Работа с форматом, где «правильно» определяется только заказчиком, — отдельная история.

4. Множитель риска.

Непрозрачные требования, отсутствие доступа к тестовой базе, «это же просто», согласования с бухгалтерией. Риск — это не «на всякий случай», это способ не разориться на задаче, где выяснилось, что правильно понял не я, а заказчик.

5. Буфер.

15–20% на непредвиденное. Без буфера первая же неучтённая мелочь превращается в переработку по чужой сдаче.

6. Итог.

Человеко-дни × ставка. Дальше — по факту.

Методика целиком

Шаг	Что делаем	Типовое значение
1	Ставка человеко-дня	Базовая ставка, от которой считаем
2	Разбивка на блоки	6 блоков: обследование, конфигурация, обмен, тестирование, документация, внедрение
3	Коэффициент сложности	1.0 · 1.3–1.5 для работ по регистрам
4	Множитель риска	По непрозрачности требований и согласованиям
5	Буфер	15–20% на непредвиденное
6	Итог	Человеко-дни × ставка

Пример: выгрузка остатков по складу в Telegram-бот для менеджеров

Блок	Оценка
Обследование	0.5 дня

Блок	Оценка
HTTP-сервис в 1С + выгрузка JSON	1 день
Бот: Python, polling, форматирование, обработка ошибок	1 день
Тестирование на реальных данных и сценарий «остатки не сошлись»	1 день
Документация и запуск	0.5 дня
Итого	4 дня × коэффициент 1.2 + буфер 15% ≈ 4.5 человеко-дня

Дальше это умножается на ставку. Обратите внимание: самая дорогая часть — не код, а тестирование на реальных данных. Почти всегда.

Что чаще всего забывают в оценке

• **Согласования.** У заказчика есть бухгалтер, методолог и руководитель. Каждый может вернуть задачу с замечаниями.

• **Обновление конфигурации.** Если меняется управляемое приложение, ваши доработки нужно будет переносить при обновлении. Это отдельная работа, которую в ТЗ не пишут.

• **Тестовая база с реальными данными.** На пустой базе всё работает.

• **Обучение пользователей.** Обычно 0.5–1 день и почти всегда «не входит в задачу».

> Методика с формулами, коэффициентами и тремя разобранными примерами — в калькуляторе стоимости доработки 1С на витрине: products.sec-scanner.pro.

Часть 3

Приёмка 1С-разработчика: что проверять, чтобы не переделывать

Момент, когда доработка объявляется готовой, — самый дорогой момент проекта. Дальше начинается приёмка, и если она идёт по принципу «открыли — работает — подписали», через месяц выясняется, что работает не у всех пользователей, данные посчитали неверно, а интеграция падает каждую вторую ночь.

Я проходил приёмку и как исполнитель, и как заказчик. Ниже — рабочий порядок проверок, который я использую.

1. Функциональность — по сценарию, а не по списку

Список вида «создан документ, работает кнопка» проверяет разработчика, а не результат. Я иду по сценарию: как пользователь выполняет рабочую задачу от начала до конца, включая отмену, повтор и отказ.

Отдельно проверяю границы: что происходит, если поле не заполнено, если документ уже проведён, если период закрыт.

2. Данные

Самая частая находка на приёмке — не логика, а данные.

• Сравниваю контрольные остатки и суммы с тем, что было до доработки, на одном и том же периоде.

• Проверяю, что документы не задваиваются и не теряются при повторном проведении.

• Смотрю на регистры: часто доработка пишет в один регистр, а отчёт читает другой. Формально всё «работает», цифры разные.

3. Интеграции

Отдельно прогоняю каждую интеграцию: обмен с сайтом, ЭДО, выгрузка в учётные системы.

• Что происходит, если контрагент недоступен? Есть ли очередь и повтор?

• Что при ошибке в середине пакета — частичная запись или откат?

• Логируется ли обмен? Без журнала любая проблема превращается в «у нас не работает, разбирайтесь».

4. Производительность

Замер на десяти записях ничего не значит. Минимум: открытие формы, проведение документа, отчёт на реальном объёме данных, выгрузка.

Отдельно смотрю на планы запросов: если в коде есть запрос, который тянет в цикле, это всплывёт на второй месяц эксплуатации.

5. Безопасность и права

• Новые роли и права: у пользователя есть ровно то, что нужно, и не больше.

• Токены, пароли и ключи не лежат в коде.

• Обновление конфигурации не ломает доработку.

6. Эксплуатация

• Документация: что изменилось, что делать при сбое, куда смотреть журнал.

• Обучение: кому и когда.

• Что делать, если разработчик недоступен. Этот пункт стоит проговаривать в договоре.

7. Сводка и замечания

Замечания оформляю в виде таблицы: что, где, как воспроизвести, ожидаемый результат, фактический. В таком виде их невозможно «забыть», и закрытие приёмки становится формальным фактом, а не спором.

Форма замечания

Поле	Что записать
Что	Краткое описание проблемы без решения
Где	Объект, экран, сценарий, роль пользователя

Поле	Что записать
Как воспроизвести	Шаги, которые приводят к ошибке
Ожидаемый результат	Что должно быть по ТЗ
Фактический результат	Что получилось на самом деле
Критичность	Блокирует работу или нет

Порядок приёмки целиком

Блок	Что проверяем	Главный риск
1. Функциональность	Сценарий от начала до конца, границы, отмена и повтор	Список пунктов вместо сценария
2. Данные	Остатки, суммы, задвоение, регистры	Формально работает, цифры разные
3. Интеграции	Недоступность контрагента, откат, журнал	Нет очереди и повтора
4. Производительность	Форма, проведение, отчёт, выгрузка	Запрос в цикле всплывёт через месяц
5. Безопасность	Роли, права, токены, обновление конфигурации	Лишние права у пользователя
6. Эксплуатация	Документация, обучение, доступность разработчика	Знания остались у исполнителя
7. Сводка	Таблица замечаний и закрытие приёмки	Замечания «забыли»

Готовый чек-лист приёмки с формами замечаний — на витрине: products.sec-scanner.pro.

Что дальше

Гайд — это три разбора, которые можно применить на следующем же проекте. Если нужно глубже — вот где искать.

Канал «1С и Telegram: разборы и практика»

Канал @onec_tg — про 1С-автоматизацию, Telegram-боты и Python: разборы, чек-листы, вакансии с гибким графиком и сообщения о новых файлах. Канал публичный, подписка без инвайта и без одобрения.

t.me/onec_tg

Витрина цифровых продуктов

Готовые рабочие документы по темам гайда: реестр интеграций 1С с Telegram, чек-лист приёмки 1С-разработчика, шаблон ТЗ на доработку 1С, калькулятор стоимости доработки и пакет КП для 1С-фрилансера.

products.sec-scanner.pro

Об авторе

Александр Ким — разработчик 1С и Telegram-ботов. Проекты и контакты: kim-cv.spru.site.

Текст подготовлен с помощью ИИ-агента и вычитан автором. Гайд бесплатный: можно скачать, переслать коллегам и использовать в работе без ограничений.