Программирование: как измерить и снизить стоимость генерации кода ИИ-агентами в 2026 году

09.11.2026 · PDF-версия статьи с сайта cms.dolinais.ru

Почему стоимость генерации стала инженерной задачей

В 2026 году типичная сессия кодинг-агента выглядит так: один промпт, десятки шагов, несколько субагентов и часы автономной работы. Публичные примеры это подтверждают: в ноябре 2026 года разбор DeepSeek Harness описывает сборку браузерной аркады на React, TypeScript и Canvas за 183 шага, 2 часа 43 минуты и 8 субагентов на подписке за $10. Когда такие прогоны становятся нормой, вопрос «сколько стоит этот код» перестаёт быть бухгалтерским — он влияет на архитектуру, процесс и выбор инструментов.

Стоимость генерации — суммарные расходы на один прогон агента: входные и выходные токены, кэшированные чтения и записи, вызовы инструментов, повторные попытки. В отличие от зарплаты разработчика, эта величина масштабируется не линейно: один неудачный шаг может стоить как десять удачных.

Из чего складывается цена сессии

Разложим прогон на составляющие, которые можно измерять по отдельности.

- **Контекст.** Каждый шаг агент заново отправляет историю: задачу, файлы, вывод команд. Чем длиннее контекст, тем дороже шаг — и тем сильнее эффект кэша.
- **Кэш.** Провайдеры дают скидку на повторно прочитанные префиксы промпта. Хорошо спроектированный агент держит стабильный префикс (системная инструкция, описание проекта) и меняет только хвост — это главный рычаг экономии.
- **Выходные токены.** Код, рассуждения и вызовы инструментов оплачиваются дороже входа. Многословные «размышления» агента напрямую увеличивают счёт.
- **Шаги и субагенты.** Каждый субагент — отдельный контекст. Восемь параллельных исследователей дешевле одного длинного цикла, но только если у них узкие задачи и общий кэш.
- **Повторы.** Неудачная сборка, падение теста, откат правки — это оплаченные шаги без результата.

  Практическое правило: если прогон не укладывается в бюджет шагов, проблема чаще в постановке задачи и обратной связи, чем в модели.

Как считать: метрики, которые стоит завести

Без измерений любые оптимизации — гадание. Минимальный набор метрик на один прогон:

1. **Токены на шаг** — вход, выход, кэш-чтение, кэш-запись. Смотрите на медиану и на хвост: один шаг с гигантским контекстом часто определяет весь счёт.
2. **Число шагов до результата** — сколько итераций потребовалось, чтобы тесты позеленели.
3. **Доля успешных шагов** — отношение шагов, давших полезное изменение, к общему числу.
4. **Стоимость задачи** — сумма по прогону, привязанная к конкретной задаче трекера, а не к «дню работы агента».

5. **Стоимость правки** — сколько стоит последующее изменение уже сгенерированного кода. Обычно именно здесь всплывают скрытые расходы.

Эти метрики удобно складывать в отчёт по проекту: тогда бюджет генерации обсуждается наравне со сроками.

Пять приёмов, которые реально снижают расход

1. Стабилизируйте префикс промпта

Держите системную инструкцию, правила проекта и структуру репозитория в неизменном виде в начале контекста. Всё изменчивое — текущий файл, лог ошибки, задача — переносите в конец. Это максимально использует кэш и снижает цену повторных шагов.

2. Дробите работу на узкие подзадачи

Один агент, который «сделает игру целиком», тащит огромный контекст. Разбиение на подзадачи с явными критериями готовности уменьшает и число шагов, и объём передаваемого состояния. Субагент получает только то, что нужно для его участка.

3. Давайте агенту быструю обратную связь

Линтер, юнит-тесты и сборка в песочнице стоят дешевле, чем шаги рассуждения. Чем раньше агент узнаёт, что сломалось, тем меньше токенов уходит на неверные гипотезы.

4. Ограничивайте бюджет шагов

Задайте лимит итераций и правило эскалации: после N шагов без прогресса агент останавливается и возвращает отчёт человеку. Это защищает от циклов, где модель «чинит» одно и ломает другое.

5. Переиспользуйте артефакты

План, схема данных, результаты исследования кодовой базы — сохраняйте как файлы в репозитории. Следующий прогон прочитает их из кэша вместо того, чтобы заново выводить рассуждением.

Что важно не потерять

Экономия не должна превращаться в самоцель. Дешёвый прогон, породивший неподдерживаемый код, обойдётся дороже: правки, ревью и отладка съедят выигрыш. Поэтому стоимость генерации стоит считать вместе с ценой владения кодом — и фиксировать оба числа в одном месте, рядом с задачами и сроками.

Инженерный вывод простой: агент — это исполнитель с оплатой по шагам. Чем яснее задача, чем стабильнее контекст и чем быстрее обратная связь, тем меньше шагов и тем предсказуемее бюджет.

Источник: https://cms.dolinais.ru/programmirovanie-kak-izmerit-i-snizit-stoimost-generacii-koda-ii-agentami-v-2026-godu