AI Dev Conf · 21 мая 2026



State of AI4SDLC: от AI-ассистентов к агентной разработке

Почему узкое место разработки переезжает из кодинга в спецификации, evals и инженерную операционную модель

План доклада

Навигация Маршрут

- Research: внедрение высокое, доверие ниже.
- SE 2.0 сдвигает узкое место поставки.
- Task management становится work graph.
- SDD и evals делают агентов управляемыми.

Часть 1

01

Что реально показывает AI4SDLC Research

Внедрение уже случилось; системный эффект появляется через проверку, процесс и платформу

AI перестал быть экспериментом в IDE

Теперь нужно управлять новой физикой разработки

Контекст Shift

Массовое внедрение

default tool

AI пишет код, тесты, docs и объяснения.

Неравный эффект

task-dependent

Кодинг ускоряется раньше поставки изменений.

Новый вопрос

verify-first

Управлять throughput, quality/risk и economics.

Почему одной метрики продуктивности мало

Исследование Метод

Читаем исследование как набор сигналов

Что измеряли

- Публичные исследования 2023-2025.
- Опрос инженеров и техлидов.

Что важно помнить

- Self-report не causal proof.
- Эффект зависит от зрелости.

Результаты Signals

Внедрение высокое, доверие ниже

Этот разрыв закрывает инженерная система

Повседневный инструмент

AI уже встроен в рабочий день инженера.

Quality needs proof

Нужны review, tests и evals.

ROI через процесс

IDE-пилот не равен SDLC-трансформации.

Кодинг быстрее, поставка не ускорилась

Главный вывод Bottleneck

Старые узкие места растут

- Review, тесты, релизы получают больше изменений.
- Слабая постановка дает неверный результат.
- Без метрик команда спорит о впечатлениях.
- Нужен управляемый контур работы.

Часть 2

02

SE 1.0 -> SE 2.0

От role-based SDLC к agent-based SDLC: меняется не инструмент, а контур поставки

Инженер управляет сквозным процессом

Два режима

Загрузка схемы...

Роль инженера становится шире

Сдвиг Leverage

Ценность — управление изменениями

Постановка

Цель, контекст, ограничения, DoD.

Оркестрация

Разложить, делегировать, собрать результат.

Валидация

Проверить поведение, риски и intent.

Команда работает по новым правилам

Не только инструменты

- Задачи пишутся для людей и агентов.
- Контекст и проверки — интерфейс.
- CI, тесты, логи — evidence.
- Нужны правила параллельных агентов.

Часть 3

03

AI-native операционная модель

Когда артефакты создаются быстрее, управлять нужно графом работы, платформой и метриками

Task management Work graph

Тикет больше не просто форма ввода

Work graph понимает контекст до задачи

Classic tracker

- Задача и статус вручную.
- Контекст в чатах, PR и docs.

Work graph

- Intake и статус частично вычисляются.
- Связаны intent, spec, PR и tests.

AI-native внедрение становится платформенной задачей

Operating model Platform

Scale требует shared capabilities

Инструкции и контекст

Repo rules, ADR, conventions, safe commands.

Проверка и наблюдаемость

Evals, CI, telemetry, cost, latency.

Защитные ограничения и экономика

Права, секреты, лимиты, approvals.

Управляйте цепочкой, не LOC

Метрики Chain

LOC — слабый сигнал

- Внедрение: кто и где использует AI.
- Throughput: lead time, review, deploys.
- Quality/risk: defects, incidents, eval score.
- Economics: tokens, infra, waiting, rework.

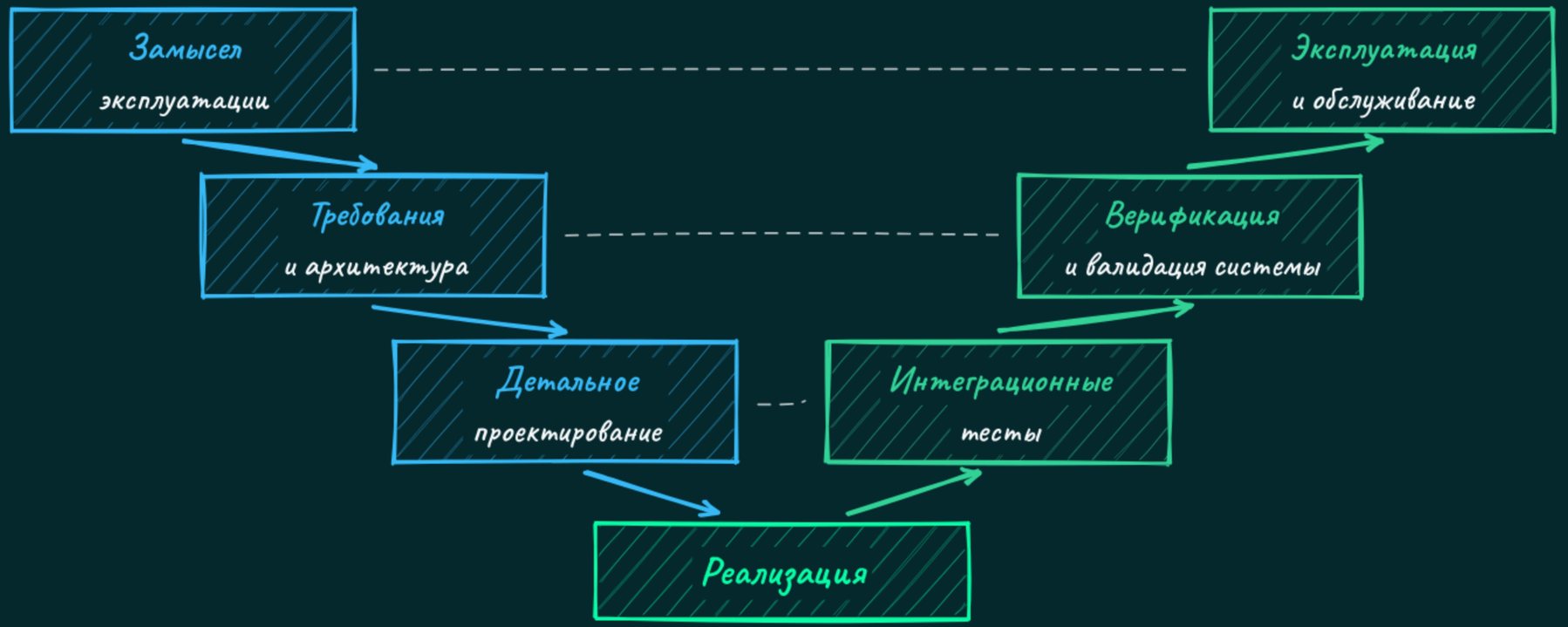
Часть 4

04

Spec-driven development

AI вернул спецификации, но теперь это не бюрократия, а рабочий интерфейс для агента

V-модель связывает требования и проверку



SDD: V-Model, но причина другая

Старые specs про трассировку; новые — про delegation

Старый spec-driven

- Требования -> дизайн -> код -> проверка.
- Сила: трассировка ожиданий.

Новый SDD

- Spec как исполняемый контекст.
- Агенту нужны intent, границы, checks.

Спецификация — это контракт с агентом

Интерфейс Human → Agent

Агенту нужен управляемый контекст

Изменение

Цель, эффект, flows, expected behavior.

Нельзя ломать

Compatibility, data, security, APIs.

Proof of done

Criteria, checks, tests, logs, rollback.

SDD: intent → proof

Документ ценен проверкой

Цепочка Flow

- Intent: зачем и какой результат нужен.
- Criteria: успех и границы регрессий.
- Plan: проверяемые шаги работы.
- Verification: tests и review подтверждают intent.

Четыре практических варианта SDD

Spec как переносимый контракт

GitHub Spec Kit

Spec -> plan -> tasks -> implement.

Kiro + OpenSpec

Структурированный и открытый spec-форматы.

Lightweight SDD

AGENTS.md, tests, PRD/RFC, checks.

Риск Reality check

Плохая спецификация ускоряет неверный код

SDD делает плохую постановку видимой

- Неясный intent усиливает неопределенность.
- Пустые criteria возвращают вкусовщину.
- Без tests/evals proof заменяет уверенный текст.
- Крупные задачи режем на chunks.

Часть 5

05

Evals для SDLC-агентов

Проверять нужно не красоту ответа, а способность агента достигать цели в воспроизводимой среде

Eval case Template

Базовый шаблон одного eval-кейса

Маленький воспроизводимый эпизод разработки

Frozen state

Зафиксированы repo, data, issue, environment.

Agent contract

Разрешенные изменения, commands, boundaries.

Hidden executable judge

Судья исполняет checks и обновляется.

Evals должны покрывать реальные типы работы

Сценарии SDLC

Одного coding benchmark мало

Product / Engineering

- Requirements: brief -> stories.
- Feature/bug: spec + regression oracle.

Quality / Operations

- Code review: merge-blocking issues.
- Tests/incidents: regression, unit, incident bundle.

Scorecard Autonomy

Автономность агента нельзя мерить одной оценкой

Карта оценки результата, пути и цены

Outcome

Задача решена, criteria прошли.

Trajectory

План, изменения, откаты, tool use.

Cost & safety

Повторяемость, latency, constraints.

Стандарт Spec + evals

Стандарт: spec, evals, CI, review

Минимальный agentic-контур

- Artifact: задачи, PR, bugs, incidents.
- Frozen state: старт и условия.
- Hidden judge: executable checks.
- Scorecard: outcome, safety, spec fit.

AI-native разработка управляется не промптами

Что забрать с собой

- AI ускоряет артефакты, а не поставку.
- SDD превращает specs в human-agent contract.
- Evals делают качество агента измеримым.
- Управляем work graph, risk и economics вместе.

- ↗aidevconf.org · ↗ai4sdlc-research.space · ↗tellmeabout.tech

Ссылки и материалы

Источники References

Источники и практики

- AI Dev Conf и AI4SDLC Research.
- Spec-driven development: Telegram-пост.
- GitHub Spec Kit и AWS Kiro specs.
- OpenAI Codex, harness engineering, evals.