

ИИ УВЕРЕН. Я НЕТ.

Как не потерять контроль над разработкой

Светлана Мелешкина

DOTNEXT



```
svetkis@dev:~$ whoami
```

Светлана Мелешкина

Lead .NET Developer

```
svetkis@dev:~$ cat skills
```

- * coding

- * vibecoding -> agentic engineering

```
svetkis@dev:~$ connect
```

- * t.me/kot_review

```
svetkis@dev:~$ dotnext run "while (true) { Speaking(); }"
```



2025:



2026: AGENTIC ENGINEERING



**КТО ОПЯТЬ ПУСТИЛ АГЕНТА В МАСТЕР?
НЕЛЬЗЯ БЫЛО ПРОСТО СКРИПТ НАПИСАТЬ?**

**ХЗ, ОН ТАМ РЕФАКТОРИЛ ЛЕГАСИ,
НАГЕНЕРИЛ ФАБРИК И ИНТЕРФЕЙСОВ, ЛОЛ**

**РЕАЛЬНО ЗАПАРИЛСЯ УЖЕ ПРОВЕРЯТЬ
ЧТО ОНИ ТАМ НАГЕНЕРИЛИ**

**ОЙ, ДА ПРОД НЕ ЛЁГ?
НЕ ЛЁГ И НА ТОМ СПАСИБО**

**Я ВАЙБКОДЕР!
БОЖЕ, КАК Я ХОРОШ
КАК МОЩНЫ МОИ ПРОМПТЫ**

Фотореференс:
Aaron Huey, National Geographic

↑ СКОРОСТЬ РАЗРАБОТКИ

↓ СТАБИЛЬНОСТЬ

+21% индивидуальная продуктивность с ИИ

-7,2% стабильность доставки (при +25% внедрения ИИ)

+23,5% инцидентов на pull request

88% команд сообщают о положительном ROI от ИИ,
но реальная трансформация редка

dora.dev/ai/roi/report



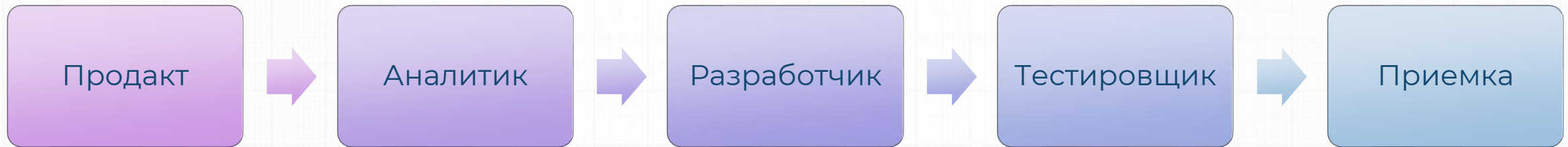
ЧТО ДЕЛАТЬ?



РАЗРАБОТКА == КОД



РАЗРАБОТКА != КОД



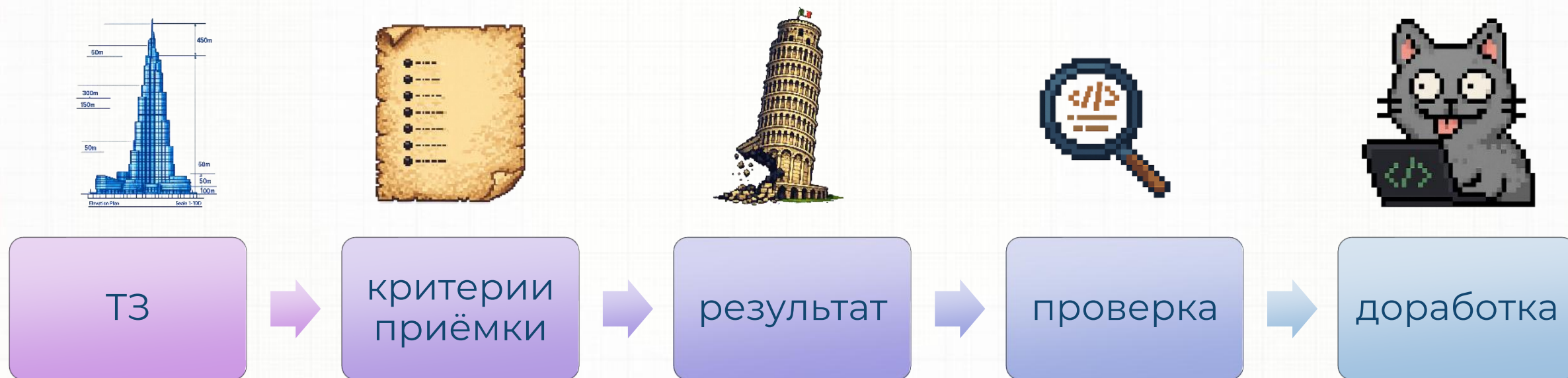
РАЗРАБОТКА ЭТО ПРЕВРАЩЕНИЕ ИДЕИ В ПРОДУКТ



OUTSOURCE



OUTSOURCE ПРОЦЕСС



АГЕНТ = АУТСОРС

Быстрый

Дешёвый

Не ходит в отпуск

Нужен строгий контроль

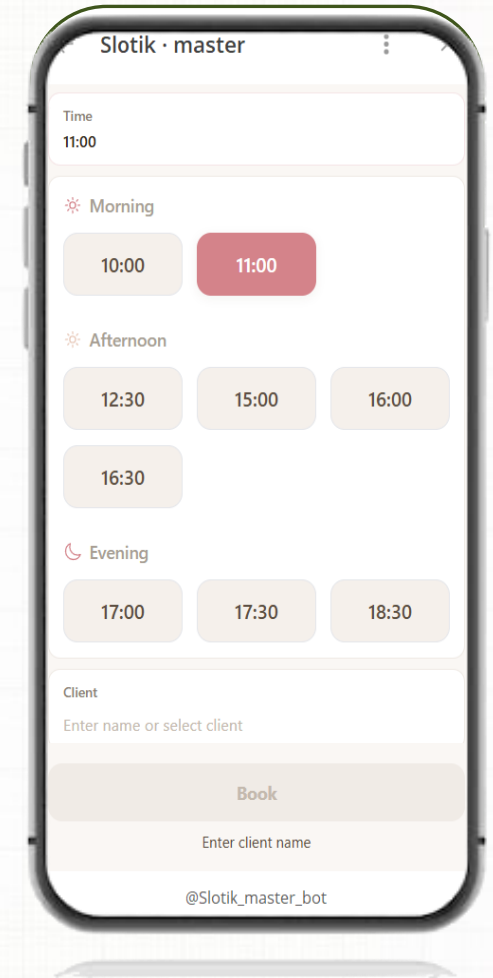
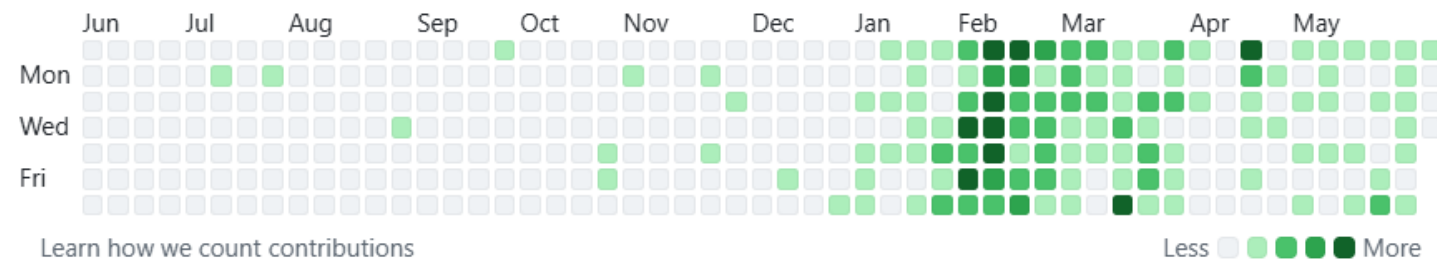


ЭКСПЕРИМЕНТ: AI SOLO DEVELOPMENT

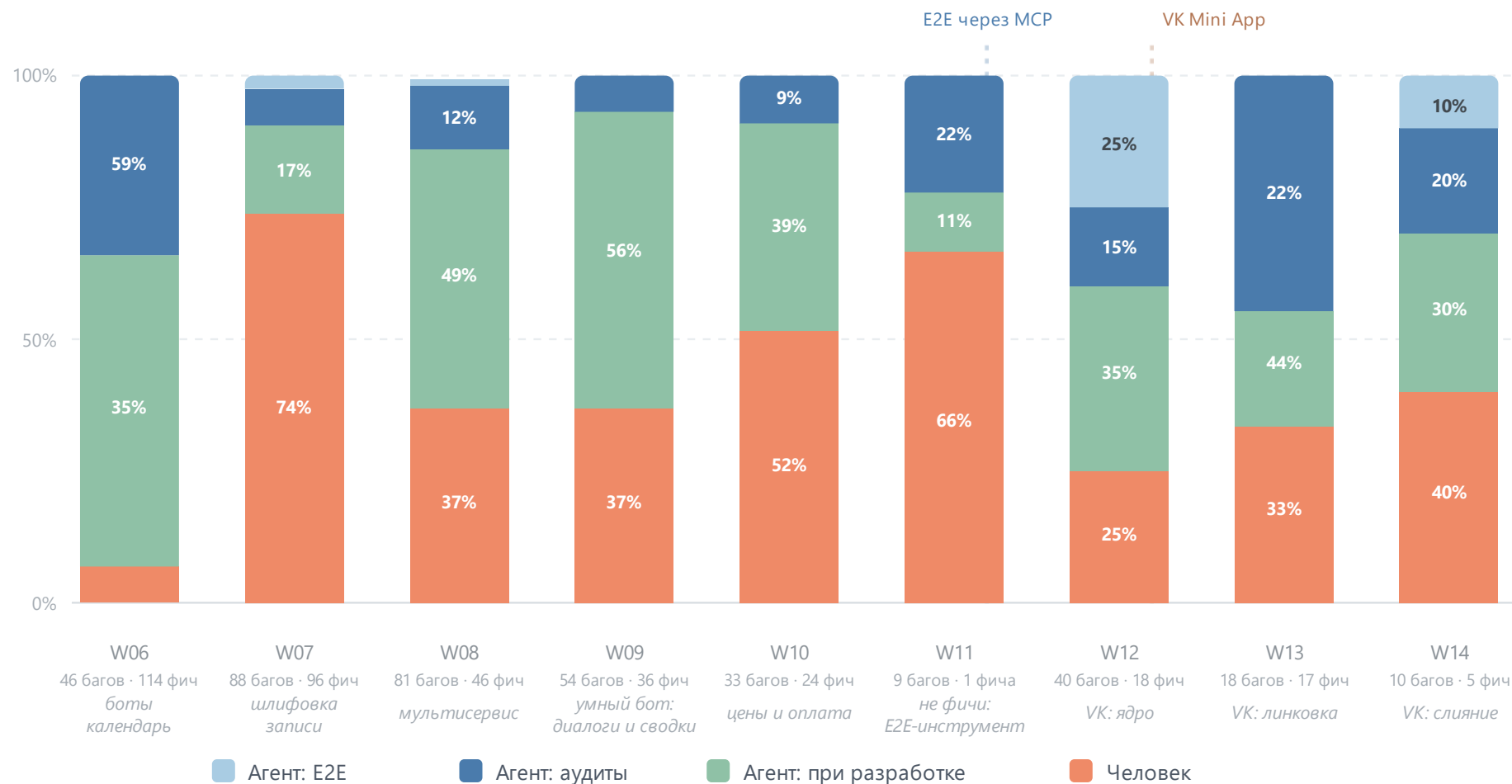
Слотик
CRM для бьюти-мастеров / Telegram WebApp

2,467 contributions in the last year

Contribution settings ▾



КТО НАХОДИТ БАГИ



ЛОВУШКИ AI РАЗРАБОТКИ



0 tests ran, exit 0

Нет запущенных тестов → нет упавших тестов.
100% success rate



```
[Test]  
Assert.True(true)
```

Code Coverage: 99%



AsNoTracking() протекшее в WritePath

- + .AsNoTracking()
- SaveChangesAsync() // молча ничего не сохранил



105x оптимизация read -> 15s write

Read latency: 2ms (x105 boost)

Write timeout: > 15s (Lock / Cache flush)



R50 красивый, max — катастрофа

Spike test: spike_2026-03-03_05-39-10.md

Endpoint	RPS	min	mean	max	p50	p95	p99
-----	-----:	-----:	-----:	-----:	-----:	-----:	-----:
global information	129.4	2.65	63.21	**4400.1**	5.94	289.0	1625
view_profile	129.4	0.09	26.14	**3376.0**	0.28	47.2	869
view_slots	129.4	2.39	37.05	**1668.0**	5.40	232.1	818



**Агент задублировал
бизнес-логику выдачи
свободных окошек для
записи**



Поломки контракта между бекендом и фронтендом



Postgres snake_case через установку имен вместо UseSnakeCaseNamingConvention()



Старые версии компонентов при новой разработке

Frontend

- React `^18.3.1` — свежий 18.x. Уже вышел React 19.
- Vite `^5.4.21` — свежий 5.x. Уже вышел Vite 6.
- TypeScript `^5.6.3`.

Если хочешь именно последние мажорные версии (React 19, Vite 6), могу поднять и пересобрать frontend — обычно там минимальные правки. Но текущий набор проверен: и `npm run build`, и ручное тестирование endpoint'ов проходят.







ФУНДАМЕНТ КОНСТИТУЦИЯ АГЕНТА



Augmented Coding Patterns

коллекция паттернов и антипаттернов

github.com/lexler/augmented-coding-patterns



LANGUAGE SERVER PROTOCOL

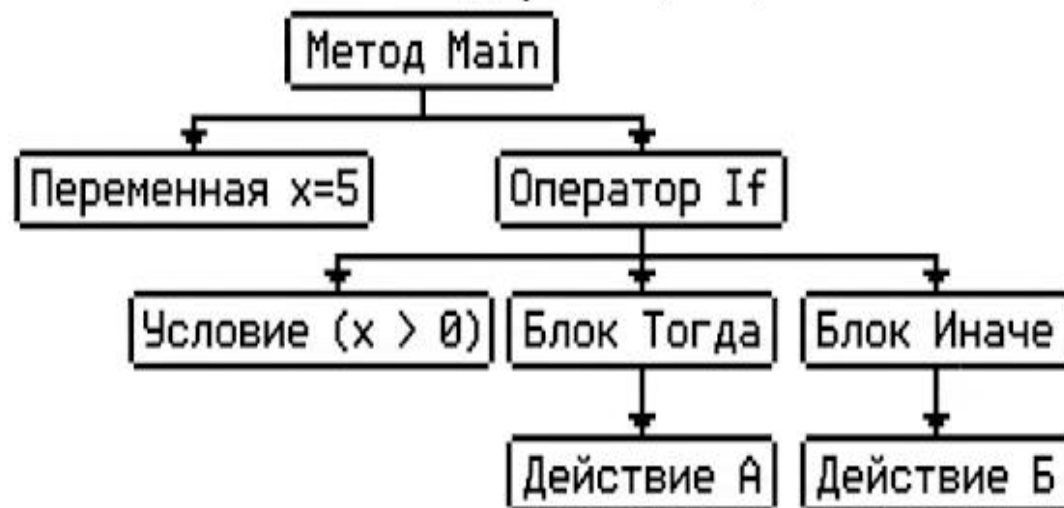
MCP: ORAIOS SERENA

github.com/oraios/serena

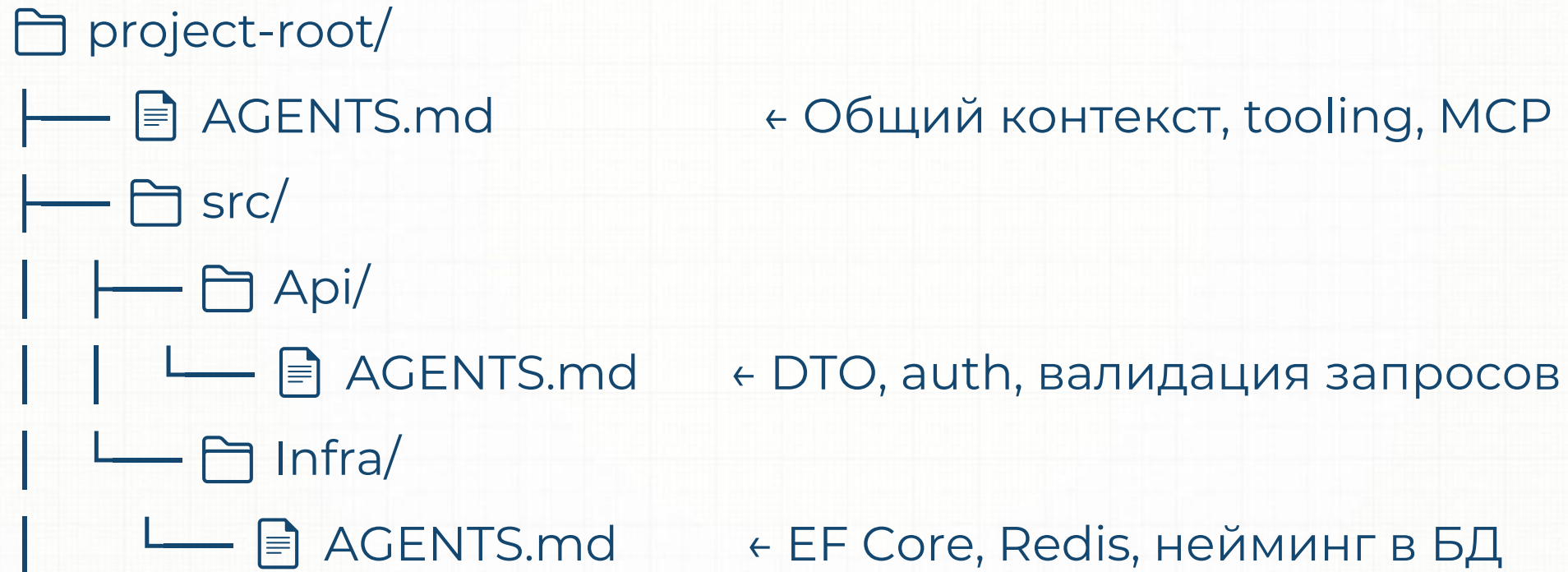
C# Код (.NET)

```
class Program {  
    void Main() {  
        int x = 5;  
        if (x > 0) {  
            // Действие А  
        } else {  
            // Действие Б  
        }  
    }  
}
```

Синтаксическое дерево (AST)



ИЕРАРХИЯ ПРАВИЛ



SEMANTIC ANCHORS

TDD, London School

Напиши изолированные тесты с моками. Фокус на поведении, не реализации. Работай снаружи внутрь. Начинай с внешнего интерфейса, тестируй взаимодействие с зависимостями...

CQRS

раздели модели чтения и записи, чтобы оптимизировать под разные нагрузки...



CONTEXT MARKERS

Эмодзи, чтобы видеть, что агент подгрузил

Из rules/AGENTS.md:

✿ — базовые правила загружены

🔍 — режим code review (diff)

☑ — режим коммита

♻ — правила перечитаны после потери контекста



ARCHITECTURE DECISION RECORDS

Документировать для агента неочевидные решения



PERMISSION TO PUSH BACK

Агенту явно разрешено сказать «нет»:

спрашивать, если инструкция противоречит правилам

подсвечивать, что не имеет смысла делать



КОРОТКИЕ ДИАЛОГИ



ГИГИЕНА АРТЕФАКТОВ

Память

AGENTS.MD

Документация



ФУНДАМЕНТ КАКИЕ ПРОБЛЕМЫ РЕШИЛИ

Context blindness

Context rot

Agent circles

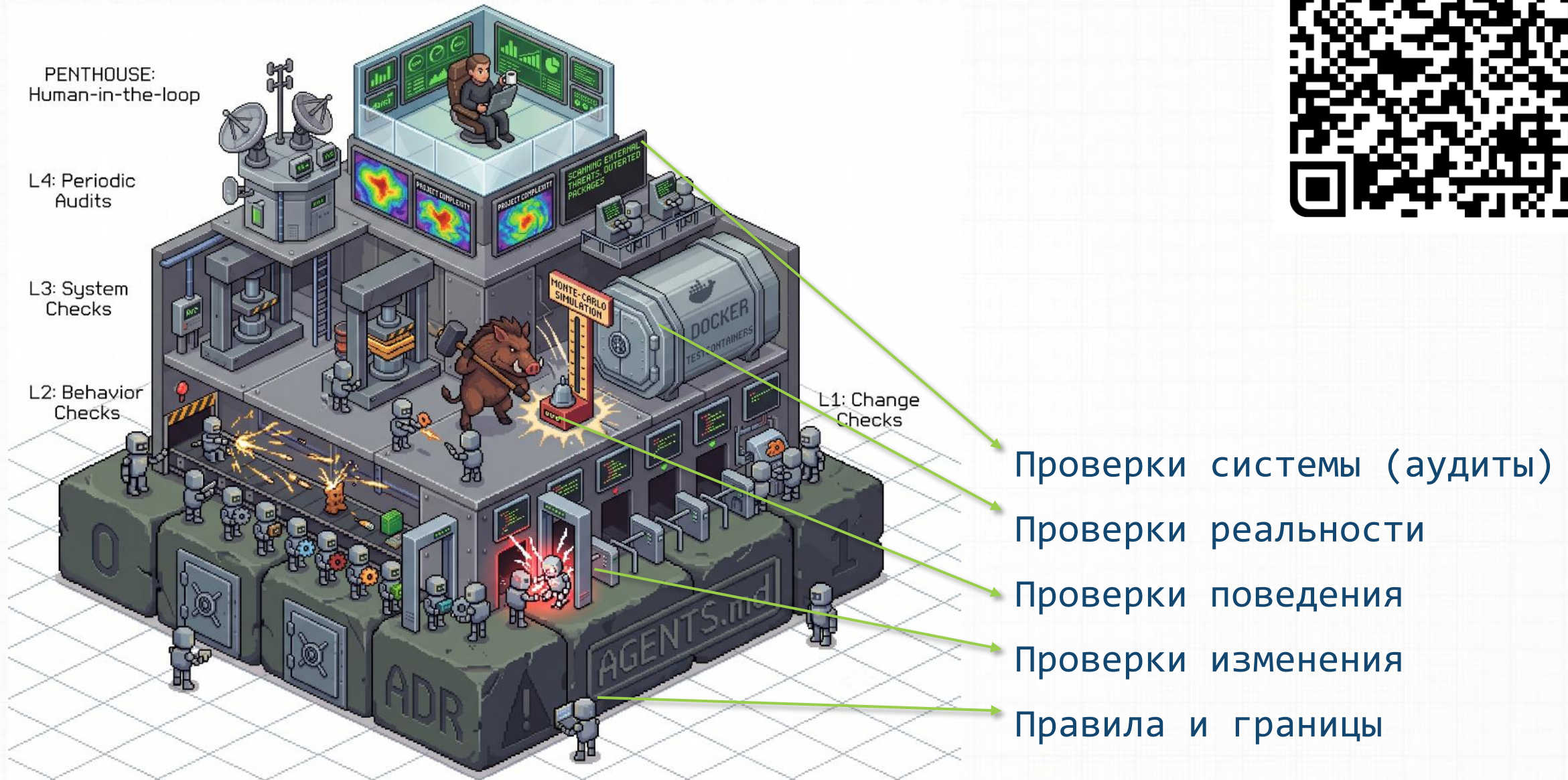


МЕХАНИЗМЫ ОБРАТНОЙ СВЯЗИ

1. Проверки изменения
2. Проверки поведения
3. Проверки реальности
4. Проверки системы (аудиты)



github.com/svetkis/agentic-acceptance-dotnet



Проверки системы (аудиты)

Проверки реальности

Проверки поведения

Проверки изменения

Правила и границы

1. ПРОВЕРКИ ИЗМЕНЕНИЯ КОМПИЛЯТОР



.editorconfig

Microsoft C# Naming Conventions

src/Directory.Build.props:

```
<EnforceCodeStyleInBuild>true</EnforceCodeStyleInBuild>
```

```
<CodeAnalysisTreatWarningsAsErrors>true</CodeAnalysisTreat  
WarningsAsErrors>
```



<Nullable>enable</Nullable>

```
public string GetUserName(User user) => user.Name;
```

```
public string GetUserName(User user)
{
    ArgumentNullException.ThrowIfNull(user);
    return user.Name ?? "Unknown";
}
```



TreatWarningsAsErrors

CS4014 Забытый await

CA2016 токен отмены не прокинут во внутренние методы

CA2000 / CA2213 забытый using того что нужно диспозить

CS0168 / CA1848 пустой catch замыливает ошибки

CS1998 псевдо-асинхронность



DEAD CODE

CS0162 Unreachable code detected

CS0219 / CS8321 Unused variable / Unused local function

CS0414 Unused private field

IDE0051 / IDE0052 Unread private member / unused private member



BannedApiAnalyzers (RS0030)

plain BannedSymbols.txt



```
# Запретить использование устаревшего класса  
T:System.IO.File  
  
# Запретить конкретный метод с сообщением  
M:System.Console.WriteLine(System.String);Use ILogger instead  
  
# Запретить метод с определённой сигнатурой  
M:System.String.Format(System.String,System.Object[])  
  
# Запретить свойство  
P:System.Environment.MachineName  
  
# Запретить конструктор  
M:System.DateTime.#ctor(System.Int32,System.Int32,System.Int32)
```



Custom Roslyn analyzers

- ✗ Сущность БД в API слое
- ✗ DbContext в контроллере
- ✗ Запрос меняет состояние (CQRS)



MEZIANTOU.ANALYZER

<https://github.com/meziantou/Meziantou.Analyzer>

MA0002 нет IEqualityComparer<string> / StringComparer

MA0006 ==/!= вместо string.Equals

MA0009 Regex без match timeout (ReDoS)

MA0011 Parse/TryParse/ToString без IFormatProvider

MA0074 Contains без StringComparison

Георгий Тормозов «Подводные камни регулярных выражений»

<https://www.youtube.com/watch?v=rIRhPoeN8tw>



Доменные типы вместо примитивов id

Newtypes / Value Objects для идентификаторов



NUGET AUDIT

КОНТРОЛЬ УСТАРЕВШИХ И УЯЗВИМЫХ ПАКЕТОВ

```
<PropertyGroup> <NuGetAudit>true</NuGetAudit>  
<NuGetAuditLevel>low</NuGetAuditLevel>  
<NuGetAuditMode>all</NuGetAuditMode> </PropertyGroup>
```

```
zsh — zsh  
  
Determining projects to restore ...  
  Restored C:\Projects\MyAwesomeApp\MyAwesomeApp.csproj (in 1.42 sec).  
  
C:\Projects\MyAwesomeApp\MyAwesomeApp.csproj : warning NU1903: Package  
'Newtonsoft.Json' 12.0.3 has a known high severity vulnerability,  
https://github.com  
  
Build succeeded.  
    1 Warning(s)  
    0 Error(s)
```

learn.microsoft.com/nuget/concepts/auditing-packages



API Snapshot Testing

Verify (Xunit, Nunit, MSTest, Tunit) SponsorCheck c v33

Storm Petrel MIT



PublicApiAnalyzers

Защита от протекания внутренней кухни из библиотеки

InternalsVisibleTo дает доступ для сборки с тестами



ЧТО МЫ ПОЛУЧИЛИ?

ПЕРВАЯ ЛИНИЯ ОБРАТНОЙ СВЯЗИ

Слой проверок за 0 токенов

Показываем как работать в нашем репо



2. ПРОВЕРКИ ПОВЕДЕНИЯ ТЕСТЫ



ПРОВЕРКИ ПОВЕДЕНИЯ

UNIT ТЕСТЫ TDD

А это точно хорошие тесты?

```
var hasAssert = methodDeclaration.DescendantNodes()  
    .OfType<InvocationExpressionSyntax>()  
    .Any(inv => inv.Expression.ToString().Contains("Assert"));  
  
if (!hasAssert)  
    ReportDiagnostic(context, methodDeclaration, methodSymbol.Name);
```



ПРОВЕРКИ ПОВЕДЕНИЯ MUTATIONS STRYKER .NET

```
// мутант выжил -> тесты врут
```

```
// код
```

```
if (slot.Capacity > booked)
```

```
// мутант
```

```
if (slot.Capacity >= booked)
```



ПРОВЕРКИ ПОВЕДЕНИЯ PROPERTY-BASED ТЕСТЫ FSCHECK

[Property]

```
public void CalculateShift_ShouldNotCrash_WithSystemExceptions(DateTime start, PositiveInt cycleDays)
{
    // Инвариант: бизнес-логика не должна падать с системным OutOfRange
    // Разработчик думает: «Смен меньше нуля не бывает, всё безопасно»

    var ex = Record.Exception(() => start.AddDays(cycleDays.Get));

    // FsCheck генерирует сотни дат, обязательно подсовывает DateTime.MaxValue
    Assert.IsNotType<ArgumentOutOfRangeException>(ex);
}
```



ПРОВЕРКИ ПОВЕДЕНИЯ КОНСИСТЕНТНОСТИ МИГРАЦИЙ EF CORE

`dotnet ef migrations has-pending-model-changes`



ПРОВЕРКИ ПОВЕДЕНИЯ АРХИТЕКТУРА + RATCHET

🍀 Fixed the failing architecture tests.

What was wrong During the `MasterViewService` refactor, the `[HotPath]` attributes on two methods were dropped when their bodies were replaced:

- `MasterViewService.GetDayTimelineAsync` — `[HotPath(PerfId = "PERF-016")]`
- `MasterViewService.GetWeekSummaryAsync` — `[HotPath(PerfId = "PERF-012")]`



ТЕСТЫ ЭТО ПАМЯТЬ О ПОВЕДЕНИИ СИСТЕМЫ

Unit тесты: это должно работать так

Architecture tests: так нельзя делать

Ratchet: не дает деградировать



ПЕРЕД СОЗДАНИЕМ PR CODE REVIEW АГЕНТОМ



3. ПРОВЕРКИ РЕАЛЬНОСТИ



ПРОВЕРКИ РЕАЛЬНОСТИ ИНТЕГРАЦИОННЫЕ ТЕСТЫ

Проверки на реальной БД

State может теряться между шагами

Баги на стыке компонентов



ПРОВЕРКИ РЕАЛЬНОСТИ

ТЕСТЫ С РЕАЛЬНОЙ БД

EF Core InMemory молча прощает то, что Postgres ломает в проде. Невалидный SQL, транзакции и FK.

```
private readonly PostgreSQLContainer _postgresContainer =  
    new PostgreSQLBuilder("postgres:17-alpine")  
        .WithDatabase("slotik_test")  
        .Build();
```



ПРОВЕРКИ РЕАЛЬНОСТИ

CHARACTERIZATION TEST

```
// 28 дней симуляции, 30 прогонов, одинаковые seed'ы  
var dzReduction = 100.0 * (avgBaselineDZ - avgSmartDZ) / avgBaselineDZ;
```

```
// симуляция дала ~61%, поэтому порог 20% – безопасный  
Assert.That(dzReduction)  
    .IsGreaterThanOrEqualTo(20.0)  
    .Because("Algo must reduce dead zones by >= 20%");
```



ПРОВЕРКИ РЕАЛЬНОСТИ НАГРУЗОЧНЫЕ ТЕСТЫ NBOMBER

Baseline: базовая расчетная нагрузка

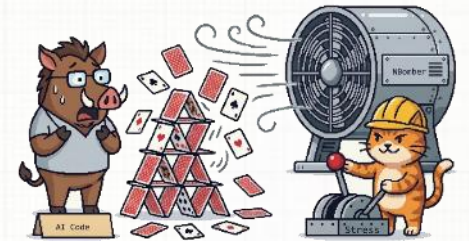
Soak: длительная нагрузка, выявление утечек памяти (`memory leaks`), пулов соединений БД и деградации со временем

Stress: поиск предела системы и проверка, как сервис ляжет

Step-up: Ступенчатый рост, проверка автоскейлинга и поведения очередей при плавном наплыве.

Spike: реакция на мгновенный всплеск (акция, пуш-уведомление), проверка троттлинга и rate limiter.

Concurrency: параллельная модификация одних и тех же ресурсов (двойные списания, овербукинг).



ПРОВЕРКИ РЕАЛЬНОСТИ E2E МСР ТЕСТЫ ПОЛЬЗОВАТЕЛЬСКИХ ФЛОУ

Тестируем боль, а не всё подряд

UI показал "Ок"? Идем в БД и проверяем бронь

Агент прокликивает стенд до живого QA

Защита от "глупых" багов



ПРОВЕРКИ РЕАЛЬНОСТИ ЧТО КОНТРОЛИРУЕМ

Проверки на реальной бд

Проверки багов на стыке

Проверки пользовательских сценариев

Проверка нагрузки



4. ПРОВЕРКИ СИСТЕМЫ

**АУДИТЫ
ЧТО НЕ ВИДИТ CODE REVIEW**



SONARANALYZER.CSHARP

КОГНИТИВНАЯ СЛОЖНОСТЬ ЦИКЛОМАТИЧЕСКАЯ СЛОЖНОСТЬ

```
public void ProcessOrder(Order order) // Цикломатическая: +1 (за сам метод)
{
    if (order.Status == Status.New) // Цикломатическая : +1 | Когн: +1
    {
        if (order.IsPaid) // Цикломатическая : +1 | Когнитивная: +2 (штраф за вложенность)
        {
            foreach (var item in order.Items) // Цикломатическая : +1 | Когнитивная : +3 (штраф x2)
            {
                if (item.Price > 100) // Цикломатическая : +1 | Когнитивная : +4 (штраф x3)
                {
                    ApplyDiscount(item);
                }
                else if (item.Price > 50) // Цикломатическая : +1 | Когнитивная : +4
                {
                    ApplySmallDiscount(item);
                }
            }
        }
    }
}
```



MEZIANTOU.ANALYZER

Метрики и дизайн

MA0051 Method is too long

MA0016 Prefer return collection abstraction

Накопленный техдолг и гигиена

MA0026 Fix TODO comments

MA0048 File name must match type name

Оптимизации

MA0106 Avoid closure by using an overload with the 'state' parameter

MA0004 Use Task.ConfigureAwait



АУДИТЫ NUGET

`dotnet list package --outdated`



ИИ АУДИТЫ УЗКИМИ ПЕРСОНАМИ

security-audit

OWASP Top 10

dba-audit

Отсутствие внешних ключей/индексов

performance-audit

Алгоритмическая сложность, лишние запросы в цикле

ux-audit

Непонятные ошибки,
странные тексты, битые флоу



ПРОВЕРКИ СИСТЕМЫ ДАЮТ НАМ ВЕРХНЕУРОВНЕВЫЙ ВЗГЛЯД НА СИСТЕМУ

Карта рисков

Контроль техдолга

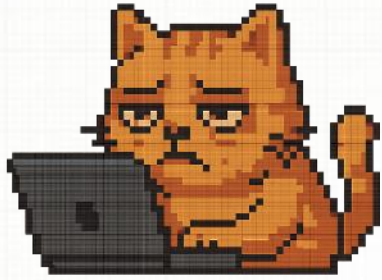




**МЫ НЕ ДЕЛАЕМ ИИ УМНЕЕ
МЫ СТРОИМ СИСТЕМУ, В КОТОРОЙ
ЕГО ОШИБКИ СТАНОВЯТСЯ
ВИДИМЫМИ**



**АГЕНТ НЕ ВРЁТ. ОН УБЕДИТЕЛЬНО
ОБЪЯСНЯЕТ, ПОЧЕМУ ТЕКУЩИЙ
РЕЗУЛЬТАТ ПРИЕМЛЕМ
(С) ИИ АГЕНТ**



HUMAN-IN-THE-LOOP



КУДА ВСЕ ИДЕТ
НОВАЯ РОЛЬ РАЗРАБОТЧИКА
ПИСАТЕЛЬ КОДА -> АРХИТЕКТОР ПРЕДОХРАНИТЕЛЕЙ

Методика Agentic Acceptance

github.com/svetkis/agentic-acceptance-dotnet

[docs/ONBOARDING.md](#) — пошаговый план

acceptance-bootstrap — скилл для сканирования зрелости и создание бэклога для применения методики

Augmented Coding Patterns

github.com/lexler/augmented-coding-patterns

SonarAnalyzer.CSharp

github.com/SonarSource/sonar-dotnet

Meziantou.Analyzer

github.com/meziantou/Meziantou.Analyzer

Светлана Мелешкина • [@svetkis](#) • t.me/kot_review

