Agentic Software Engineering @ Parloa

Parloa

Pedro Castillo, Staff Engineer

Agent Reliability and Evaluations

- Animal Handling!
  Pythons 🐍, SLOths 🦥, Elephants 🐘 or Dolphins 🐬
- Sweep problems under any rug 🧹
- Forage *MangoDBs* 🥭
- Make queries go brrr! 🏎️
- Make all your streams come true 🏞️

Valencia, VE — Buenos Aires, AR — Berlin, DE

The leaders of this era will provide a personal AI agent for every customer

AI Agent Management Platform

AI Agent Lifecycle

Customers

Channels

- PHONE
- MESSAGING
- CHAT
- CLICK-TO-CALL
- MULTIMODAL

CX Platforms

CCaaS

more...

MESSENGERS

WEBSITES & APPS

more...

AI AGENT

1. DESIGN AND INTEGRATE
2. TEST AND ITERATE
3. DEPLOY AND SCALE
4. MONITOR AND IMPROVE

INTELLIGENCE

Azure | Google | OpenAI | BYO STT / TTS / LLM

Ecosystem

OTHER AGENTS

- AI AGENT
- HUMAN
- AGENT ASSIST

INTEGRATIONS

salesforce | SAP | servicenow | zendesk | UiPath | VERINT | Microsoft | celonis

Agenda

1 AI adoption jouney

2 A look into Parloa’s Kitchen

3 Lessons against the *slop*

4 What’s next

Chapter 1

AI adoption journey

Our Journey

DEC 2024

Cursor

Cursor, AI code editor, is evaluated and introduced to developers at Parloa.

FEB 2025

Agent Mode

Cursor's Agent Mode gets traction and adoption across engineering teams.

APR 2025

Accelerator Week CursorMeister

Tech org AI coding training and challenge using Cursor. A data scientist and a product analyst did the fastest successful submissions of a microservice.

MAY 2025

Parloa AI-SDLC

Curated set of .cursorrules and MCPs shared across teams

JUL 2025

Claude Code

Claude Code, the agentic CLI for software development, is adopted by engineering teams at Parloa.

AUG 2025

BugBot

BugBot, the code review bot, is enabled by default in all code repos.

OCT 2025

Parloa Claude Kitchen

Parloa's Claude Code plugin marketplace with curated and experimental plug ins.

Show me how you cook

Weekly event where developers share their workflows, learnings and challenges with live demos only.

NOV 2025

Accelerator Week From AI pair programming to AI-enabled engineering

Tech org workshop on moving from synchronous AI collaboration to asynchronous AI delegation.

JAN 2026

JarvisOS for PMs

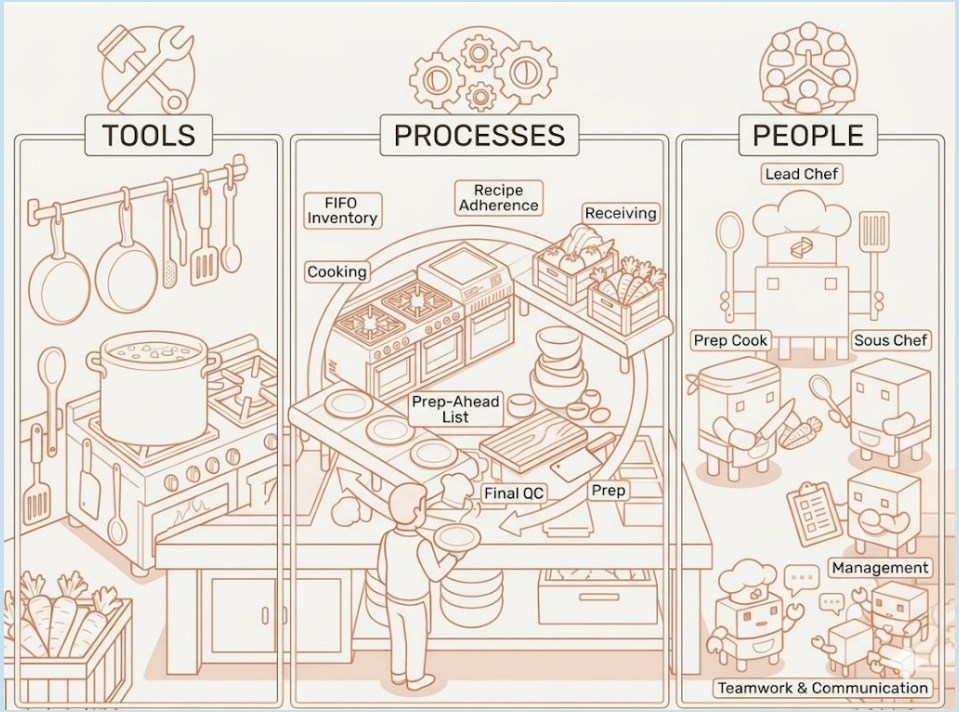
Claude Code skills and settings for Product Managers, focused on early phases of the product development lifecycle: research, discovery and validation.

TODAY

Chapter 2

A look into Parloa's Kitchen

Good software is made of...



Tools

Cursor

Access to multiple models.
Best autocomplete – period.
Agents inside the IDE.

Claude Code

The most programmable agentic CLI.
Scriptable: `claude -p`
Vibrant community.

The Classics

Familiar tools, durable outcomes.
Trusted systems of record.
Collaboration and handoffs.

But we also build tools for our tools

Parloa's Claude Kitchen

Plug-in ecosystem for Claude Code

Give Claude superpowers!

Claude Kitchen contains a set of plug-ins & skills (we call them *recipes* 👨‍🍳) to enhance the basic Claude Code experience.

We leverage Claude's plug-in and marketplace features to package and share specialized features to Claude Code.

From generic to opinionated expert

On demand expertise

The recipes is how we encode best practices and how we do things ***the Parloa way***.

It gives Claude our opinions and reasons to favor some patterns over others.

Actionable guidelines

Guidelines without headaches!

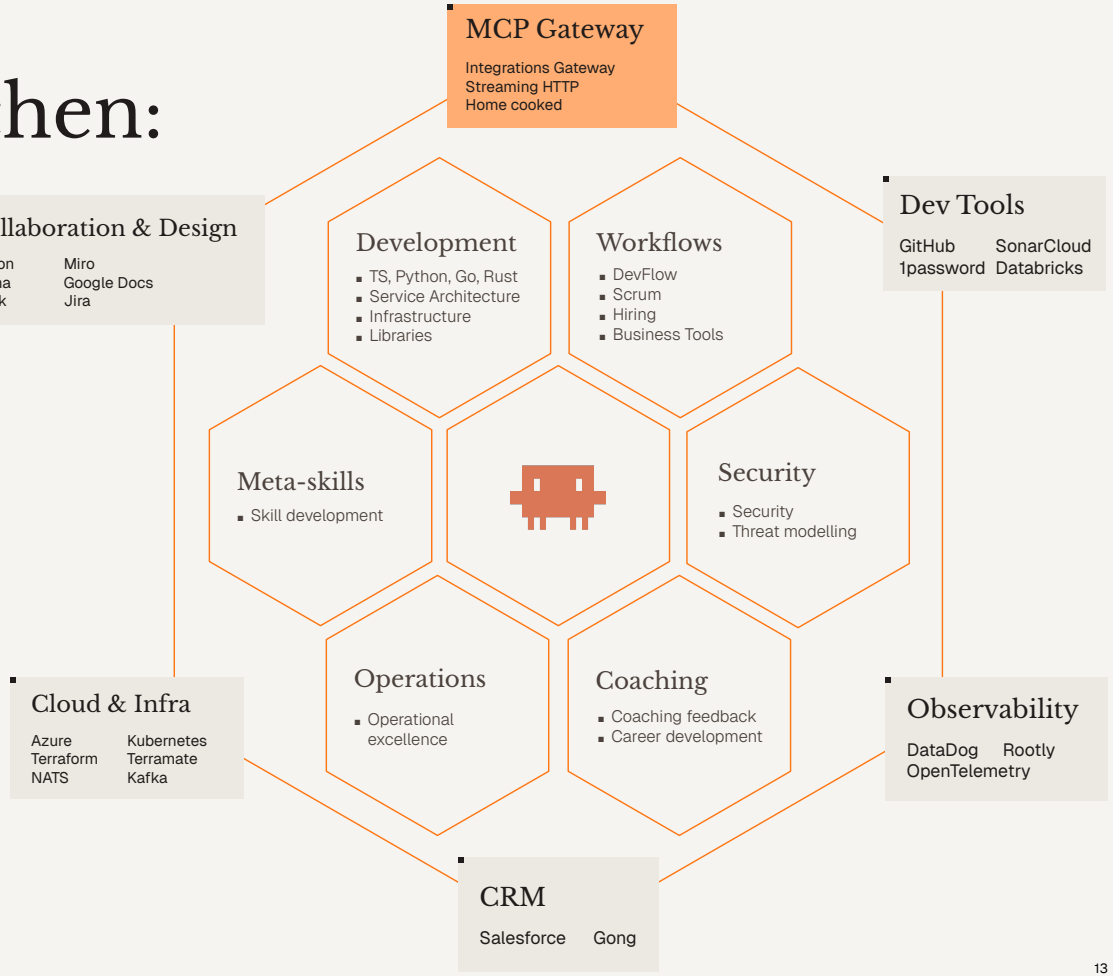
It is typically hard to enforce and review the adherence to Parloa's guidelines.

Now we can enhance Claude to always remember them and use them whenever needed, making our guidelines truly actionable.

Parloa's Claude Kitchen: Architecture

Overview

- 21 Plug-ins
- 67 Skills
- 500+ Tools
- Skill simulation and evaluation
- Verification steps



Processes

Prototype

Prove the concept.
Could this work?

Plan

Choose the path.
How will we do it?

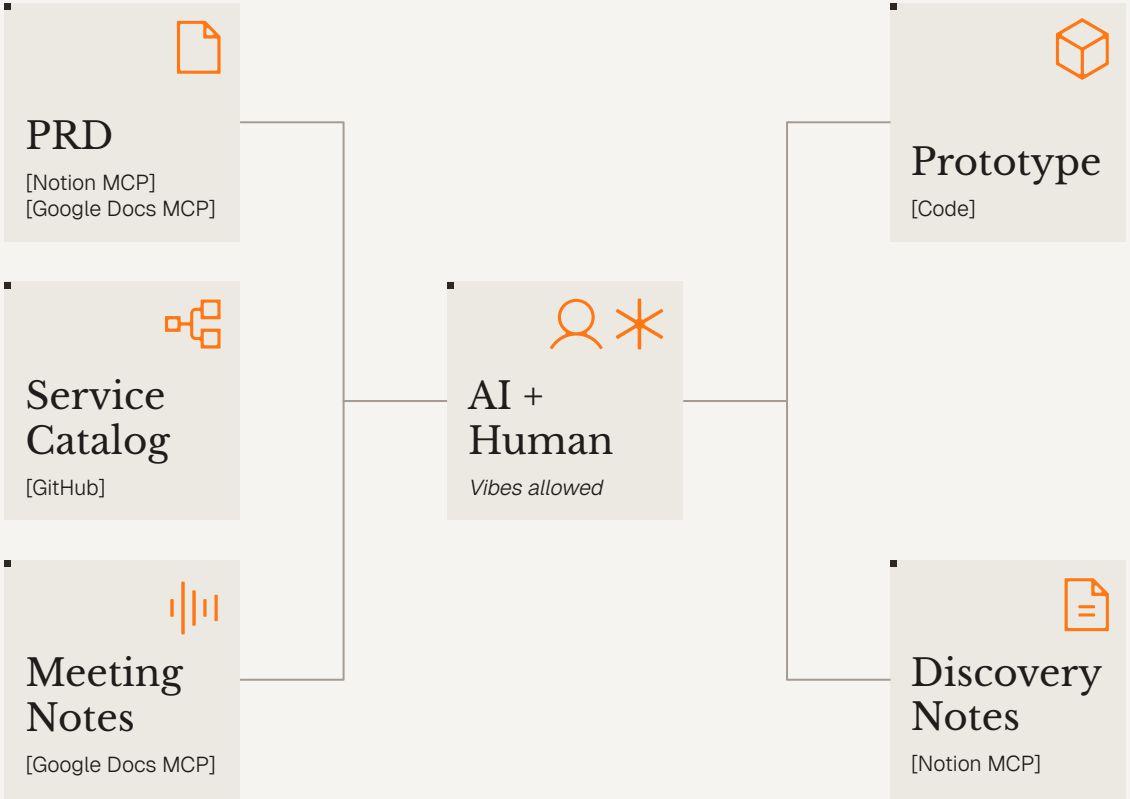
Implement

Deliver the value.
Did we make it real?

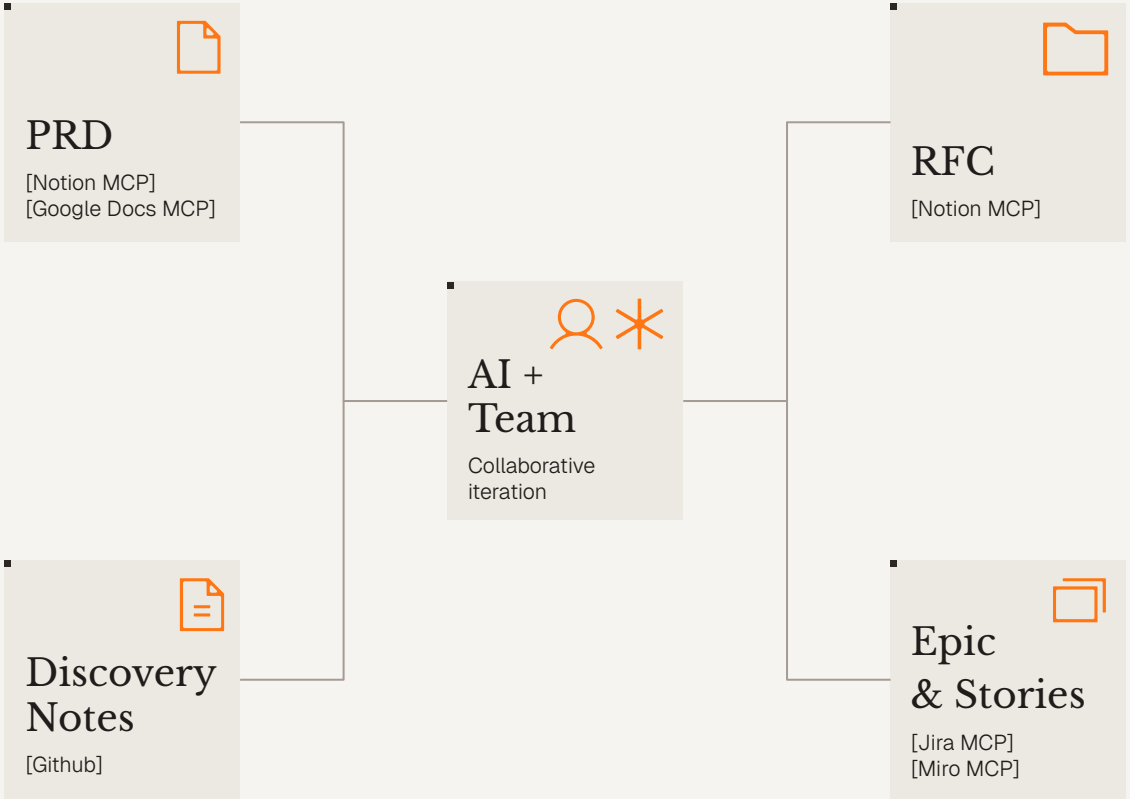
Maintain

Improve and sustain.
How do we keep it good?

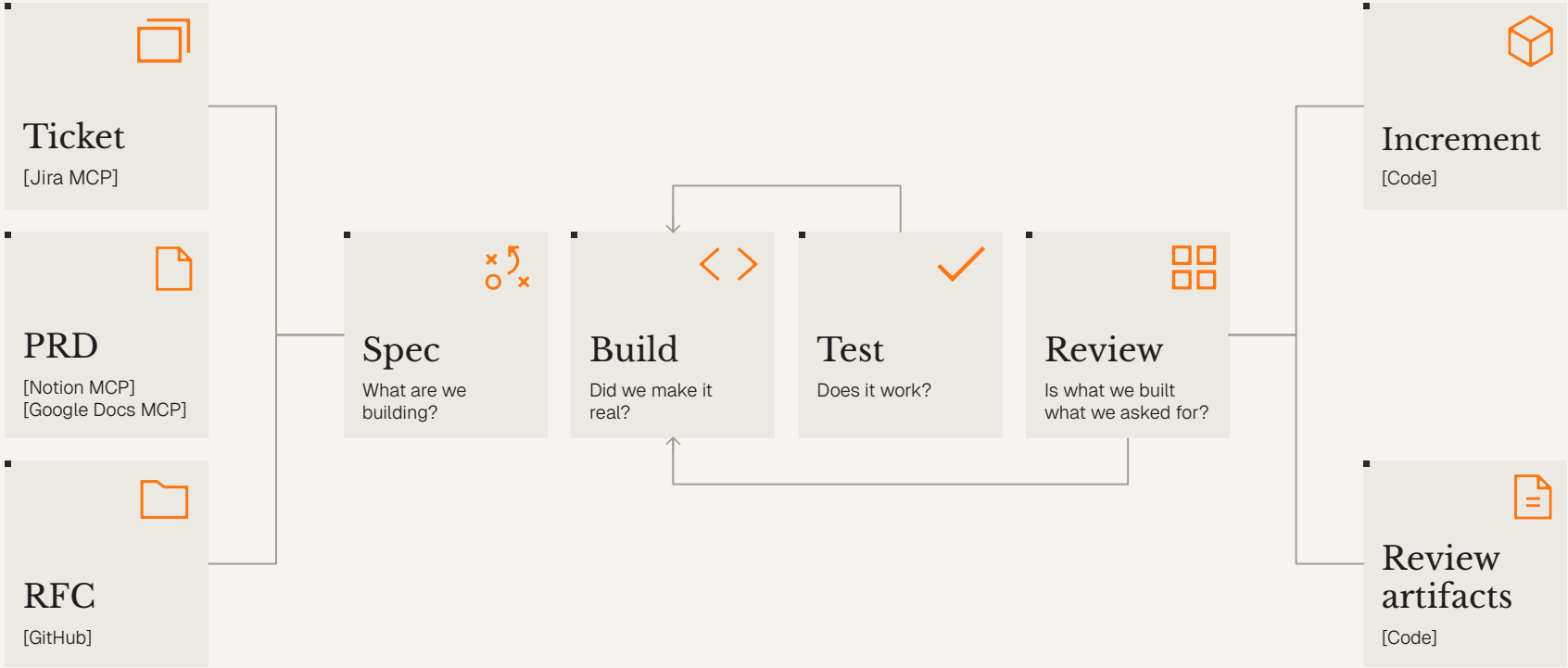
Processes: Prototype



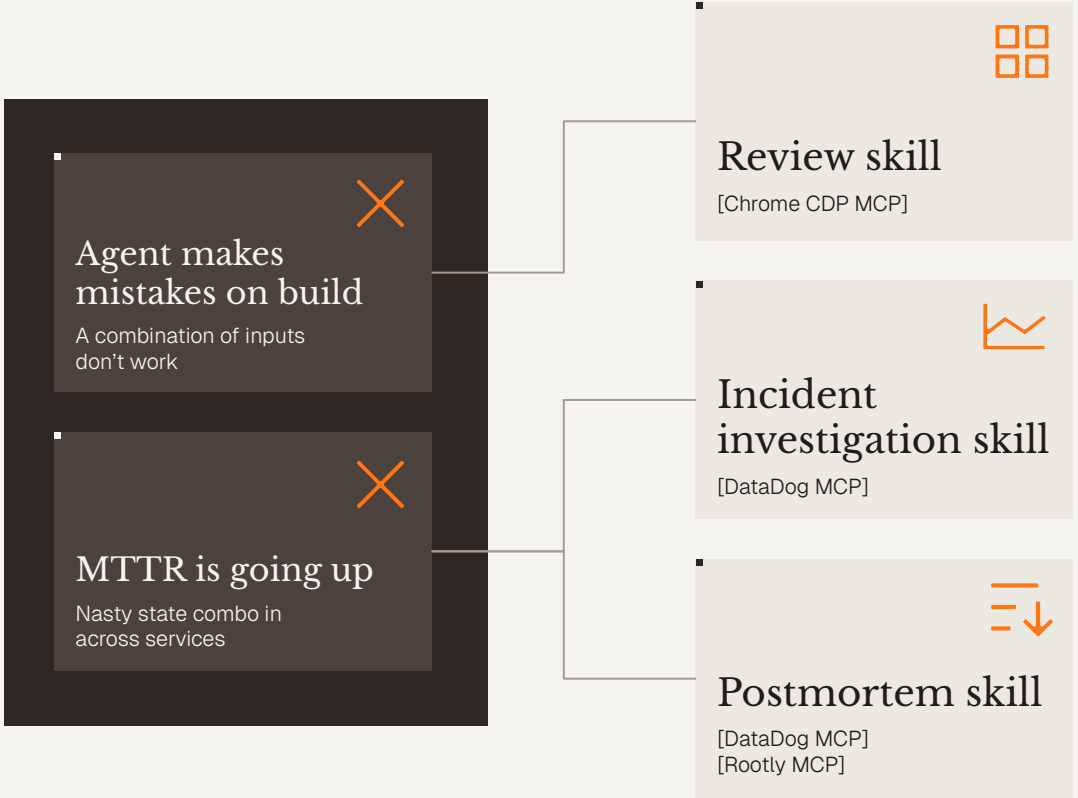
Processes: Plan



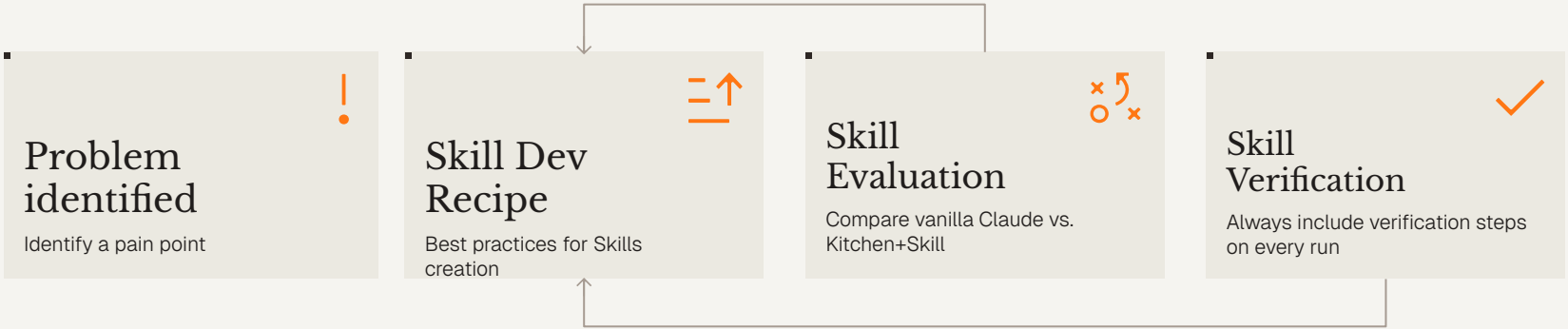
Processes: Implement



Processes: Maintain



Processes: Skill Development



People

Mindset

Product-minded architects

"We need to become product-minded architects managing the production of code"

Bottlenecks

Time and effort to Cognitive load

Everything is changing fast. Work intensifies managing parallel workstreams. Work smarter, not harder.

Collaboration

Pair programming to Pair conceptualizing

Now we all need to learn to collaborate with a new stakeholder: agents.

Learning

New tools, new Skills

AI Skills require experimentation in a encouraging and fun environment.

People: Testimonies

Every week, I use CC to check my interview from calendar and write all the prep docs.

Principal Engineer, Hiring Plugin

Others got surprised when they got a Slack message for PR review while I was presenting in a meeting.

Staff Engineer

Now we can focus on improvements and worry less about preparing ceremonies.

Engineering Manager, Scrum Plugin

Instead of refactoring, I rewrote the whole library keeping the API and tests, now runs faster with 90% code reduction.

Senior Engineer

I made a FinOps agent to retrieve, sort and upload invoices, it save me hours.

VP of Engineering

Lessons against the *slop*

Lesson #1: Empower experimentation

Interesting quote: devs should experiment with GenAI

From 53:30:

> Gergely: "I wonder if we're going back to discovering things that we you were popularizing in the 2000s."
>
> Kent: "People should be experimenting. Try all the things, because we just don't know.
>
> **The whole landscape of what's 'cheap' and what's 'expensive' has all just shifted.** Things that we didn't do because we assumed they were going to be expensive or hard just got ridiculously cheap. Like, what would you do today if cars were suddenly free? Okay, things are going to be different, but what are the second and third-order effects? Nobody can predict that! So we just have to be trying stuff."

Kent Beck, The Pragmatic Engineer, June, 6th 2025

Provide space and tools to learn

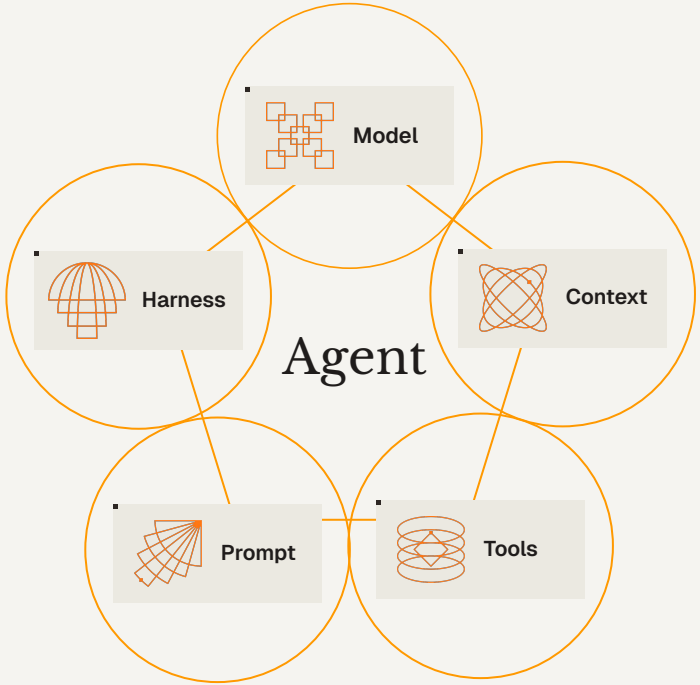
Agentic AI requires practice, but onced the you pass the initial resistance, it becomes fun and rewarding.

Develop your intuition

Instead of saying "AI can't do that", give it a try.
Pay attention to what works, learn from what doesn't.

Take note of what doesn't work and use it to judge the next model.

Lesson #2: Master the primitives



Consider your agent's perspective

Your agent is brilliant but amnesic. It is ephemeral, it has no memories. If you want your agent to perform like you, it needs your perspective.

On every prompt, start asking yourself:

With my agent's model, context, prompt, tool and harness , is it possible to safely complete the task I've given it?

Study the harness, tinker like a dev

We are getting more and more **programmable** AI tools.

Explore features, see how they compose, create automations.

Simply speaking, Agents are CLI tools, use them in similar places.

Lesson #3: Service templates and tooling

Standardize structure and patterns

Agents have the tendency to replicate what already exists.

Create strong templates with code organization and architectural patterns.

Guardrails: local and CI

Be strict at the beginning and offer a solid foundation, then watch the codebase go in auto-pilot.

Include static analysis, security and quality tools.

DevX + AX

Agents are your new users. Invest in Agent Experience (AX). There is a good overlap between DevX and AX.

Provide safe access to the same tools.

Lesson #4: Feedback loops

Close the loop

Create closed-loop systems where agents can validate their work. This is the single most leverage thing you can do.

Use the pattern:
Request → Validate → Resolve

Focus on workflows

We all want the super agent that can do it all with minimal input. We know that it doesn't exists.

As engineers, we operate one step at a time, each step requires different information and a different approach. Encode that for your agents.

Design the environment

Software discipline has shifted **from writing code to creating the scaffold that supports it.**

Our core challenge is now designing the environments and feedback loops that enable agents to build complex and reliable systems.

Lesson #5: Code reviews need to change

A good code review requires attention and time – You have neither.

Staff Engineer

Spec review

Review the spec, then the diff.

Shift from line-by-line reviews to stronger checks and clearer intent.

Proof review

Remember the Review step: Is what we built what we asked for?

Make the agent prove it!

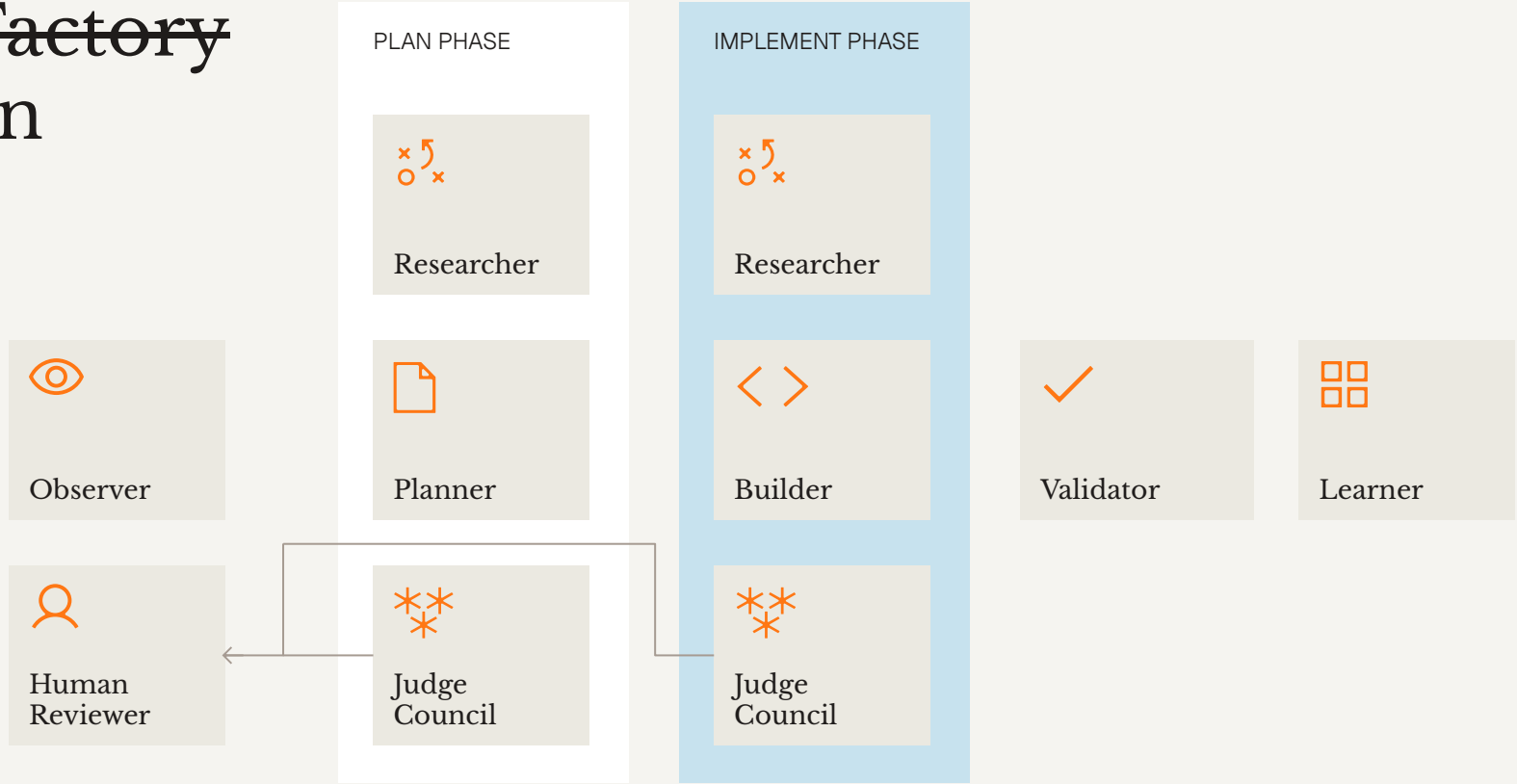
Review bots

Bots can focus on line-by-line reviews.

They enforce guidelines, policies, security and classify risk.

What’s next

Dark ~~Factory~~ Kitchen




We are hiring!
Come and build with us!
parloa.com/careers
Parloa

Any questions?

Are you ready to build meaningful relationships with your customers?