



Compute AI Offsite: Forging the Future of AI-Assisted Engineering



Rik Kisnah

Consulting Member of Technical Staff

4 min read

Compute AI Offsite: Forging the Future of AI-Assisted Engineering

December 9–11, 2025 | Seattle, WA

Last week, OCI Compute brought together IC5s and IC6s from across the organization for a three-day AI offsite in Seattle. This was not a typical corporate gathering. It was a deliberate effort to accelerate AI adoption within Compute engineering and to short-circuit the traditional onboarding curve for tools that are already reshaping how we work.

This was a working offsite—hands-on, opinionated, and grounded in real Compute problems.

Historical Context: Standing on the Shoulders of Giants

This offsite echoed several pivotal moments in our history:

- **RIC L11 (RDMA / H100 QFAB bring-up)**
Led by Jag Brar, a small, focused group tackled one of our most complex hardware integrations.
- **GB200 Seattle Offsite**
Cross-functional collaboration delivered one of the most complicated SKUs in record time.

Both demonstrated the same principle: when the right people are brought together with a clear mission, execution accelerates. The AI offsite followed that same playbook.

The Mission: AI Tools for Compute Engineering

Rahul Chandrakar, Architect working for Tyler, led the event using a structured three-component approach:

1. **Theory**
Understanding what AI agents are and how they work.
2. **Business Cases**
Identifying where AI tooling delivers measurable engineering value.
3. **Hands-On, Real Problems**
Applying AI directly to live Compute codebases and tickets.

The focus was on onboarding tooling developed by Greg Pavlik's team:

- **OCA (Oracle Cloud Assist)**
- **Cline**
- **ocaider (agentic client)**

Rather than abstract demos, teams worked on real repos, real logs, and real incidents.

Key Innovation: Prompt and Context Engineering in the Codebase

A clear paradigm shift emerged:

Markdown is now a first-class citizen in our repositories.

Because LLMs operate on tokens, structured context becomes critical. Practically, this means:

- **Intake documents** live in repos as Markdown—not Confluence
- **Design documents and ECARs** are source-controlled artifacts
- **Code is generated from structured documents**, not the reverse

For engineers accustomed to a Jira → Confluence → Code workflow, this represents a fundamental change. Documentation is no longer an afterthought—it becomes the **prompt that drives implementation**.

What We Saw in Action

Several demos stood out:

- **Ticket Analysis**
Models identified recurring patterns across Compute tickets with ~90% accuracy, trained through early human feedback. Additional demonstrations by Madhav (Bhaswati's org) on using AI with RAG for ticket analysis were particularly effective.
- **Legacy Bug Fix**
A 7-year-old typo in the Compute Control Plane ("Sheperd" vs. "Shepherd") was diagnosed and fixed in ~15 minutes. While not critical, it served as a clear proof point for accelerating long-tail cleanup.
- **Design-First Workflows**
Teams that began with structured HLD/LLD documents and Mermaid diagrams consistently achieved better outcomes than those that jumped straight into code.

What Worked Well

- Design-first workflows (HLD/LLD before implementation)
- Mermaid diagrams for flow visualization

- Prompt templates to constrain scope
 - Logs, code, and tickets available in a single shared context
 - Explicit requests for effort estimates and change breakdowns
 - A repeatable structure shared by Rahul that will be adopted across repos
-

What Did Not Work (Yet)

- Context drift during long sessions
- Large files (especially YAML) overwhelming tools
- AI-generated code without human validation
- One-shot, end-to-end API generation

Key takeaway: This is **AI-assisted engineering**, not AI-driven engineering. Humans remain the validators.

Breakout Groups

Participants split into domain-focused breakout sessions:

- Compute Control Plane
- Hardware (ILOM / PUL / TRS)
- Hypervisor / OCA / VMIDP
- Network Auto Ops
- Linux

Each group applied the same tooling to different problem spaces, building muscle memory that carries back into day-to-day work.

Leadership Alignment and Support

This initiative succeeded because leadership made it a priority.

- **Donald** and **Tyler** pushed for a top-down commitment.
- **Rahul's** technical leadership—as Tyler's architect and a frequent CCAR/ECAR driver—enabled execution.

During a working lunch, **Sudha** encouraged AI2 to define concrete KPIs with M5/M6 leadership and demonstrate measurable progress by January. The mandate was explicit: **show results, not just potential**.

Next Steps: The AI2 Compute Proposal

The proposed path forward:

1. Review with **Sudhir**
2. Review with **IC5s**
3. Review with **Avnish**

Operating principle: Start *very* small. One SDE. Incremental impact. Expand based on results.

Experience at Amazon and Microsoft shows that broad, rapid transformations often stall under competing priorities, cultural friction, and unclear KPIs.

A joint **Compute + AI2 committee** has been formed and will meet monthly to share progress, metrics, and best practices.

Bottom Line

Expectations going in were modest—many offsites devolve into corporate theater. This one did not.

It remained technical, practical, and engaging for all three days. It reinforced confidence that OCI Compute is meaningfully accelerating internal AI adoption—closing the gap with what Amazon, Microsoft, and Anthropic already practice internally.

As Joella Matthias (AI2 Compute) reminded us repeatedly:

AI will not replace ICs—but it will absolutely replace ICs who refuse to adapt.

The offsite is over. The work begins now.

Acknowledgments

Kudos to **Jen** and the TPM team for coordinating logistics, and to all participants for leaning in and making this a working offsite. Great to reconnect with familiar faces and meet new colleagues.

Side note: AJ and I also navigated a non-AI disaster when our homes flooded in Maple Valley—a reminder that the real world still throws curveballs that even the best models cannot predict.