Part IV — The Breaks Between Eras

Chapter 19 — Case Study: From PC Software to the Web (Thing → Network)

This is the first big modern "thing → network" transition on the clock:

- **PC (12 p.m.) → Internet (6 p.m.)**
- From **local, owned software** to **remote, shared services**.

If you strip the nostalgia out of it, this is what actually changed — and what you should be watching for as we go from AI to robots.

Life at 12 p.m.: PC Software as the Center of the Universe

In the PC era, the world made sense like this:

- You bought a machine.
- You installed software on it (from disks, then CDs, then downloads).
- Your files lived locally.
- Your workflows were:
    - Word, Excel, Photoshop, AutoCAD, local databases, custom apps.

Key assumptions:

- **Distribution**: software is a product you ship and update infrequently.
- **Control**: the user controls the environment; IT can lock it down, but the machine is "theirs."

- **Value**: the thing people pay for is the **program itself**.

This is pure 12 p.m. energy: concentrated, local, owned.

The Network Sneaks In: Email, Intranets, Browsers

The early web didn't arrive like a revolution. It snuck in as:

- Email clients.
- Corporate intranets accessed via Netscape/IE.
- FTP, telnet, bulletin boards.

Then suddenly:

- "Instead of shipping software, what if we just host it and let people access it through a browser?"

That one shift changed everything:

- No install process.
- No per-machine updates.
- No "version hell."
- Continuous delivery, not discs in envelopes.

We moved from **programs on your machine** to **services you reached over a network**.

The clock hand moved from 12 toward 6.

The Core Break: From Files and Installs to URLs and Sessions

The decisive moment was mental, not technical:

> The unit of work stopped being “a file opened by a local program”
> and became “a session on a remote service.”

Once you accept that:

- Identity = login, not local profile.
- Persistence = server-side state, not local files.
- Distribution = URLs, not physical media.

You get:

- SaaS instead of shrink-wrapped software.
- “Sign up” instead of “install.”
- Subscriptions instead of boxed licenses.

What died for anyone who clung to it:

- The idea that your software lives *primarily* on the user’s machine.
- The notion that release cycles are annual events.
- The belief that offline is the default, and online is the exception.

What PC-Centric Players Got Wrong

The teams that struggled tended to make the same mistakes:

- **They treated the web as an add-on, not the new default.**
  - “We’ll add web syncing later.”
  - “We’ll put our documentation online.”
  - But the core value stayed local.
- **They assumed local control was a permanent requirement.**
  - “No one will trust mission-critical work to a browser.”

    - “Enterprises will never accept their data in someone else’s data center.”
    - Many did… until convenience won.
- **They optimized for the wrong constraint.**
    - They kept designing for CPU and disk, while the world started designing for bandwidth and latency.

By the time they woke up, the center of gravity had already moved. The browser had become the main stage.

What the Web-Centric Winners Understood

The winners leaned into three ideas early:

1. **The browser is the runtime.**
    - Treat it as the target platform, not a thin client.
    - Assume you can build serious tools with HTML+JS.
2. **Identity is now server-side.**
    - Accounts, profiles, permissions — all anchored in the cloud.
    - The same “self” follows you from machine to machine.
3. **Shipping never stops.**
    - Continuous updates instead of big-bang releases.
    - Features, fixes, and experiments pushed from the center.

This is pure 6 p.m.: diffusion, participation, accounts, users, subscriptions.

The Takeaway for Robot Noon

Why does this matter for robots?

Because **you’re living through the equivalent break again**:

- PC → Internet was **Thing → Network**.
- AI → Robot will be **Network → Thing**.

Anyone who treats robots as “just another endpoint for the same network behavior” is making the PC-software mistake in reverse.

The lesson from this case study:

> When the substrate changes (from local to network, or network back to embodied), the unit of work, the business model, and the UX pattern will not survive intact.

Design for the new substrate, not as a plugin to the old one.

Quick Exercise

Pick a classic desktop-era product (word processor, graphics tool, accounting software) and answer:

1. What changed about it when it moved to the web?
2. What assumptions died (about install, files, updates, pricing)?
3. Now map that same pattern: what assumptions about **AI today** are likely to die when it moves into robots?

Write it out. You’re training your pattern detector.

Chapter 20 — Case Study: From Web to Smartphone (Network → Thing)

This is the next big break:

- **Internet (6 p.m.) → Smartphone (12 p.m.)**

- From **network you visit** to **thing you carry**.

Most people remember this as “the rise of mobile.” What matters for our clock is what actually flipped underneath the buzz.

Life at 6 p.m.: The Web as the Default Interface

Before smartphones took over, reality looked like this:

- You sat at a **desktop or laptop**.
- You opened a **browser**.
- You went to **websites**:
    - portals, blogs, forums, search engines, webmail, online shops.

Key assumptions:

- Screen: big, horizontal, mouse-driven.
- Input: keyboard, precise cursor.
- Sessions: bounded — you “go online” for a while, then stop.
- Identity: lots of separate logins, cookies holding things together.

The web was the place you went. Your device was a terminal. Pure 6 p.m.

Enter the Smartphone: A New 12 p.m. Thing

Then this new artifact showed up:

- Small vertical screen.
- Always with you.
- Multi-touch interface (finger, not pointer).
- Battery constraints, radio constraints, sensors (GPS, camera, accelerometer).

It was clearly a **thing**:

- You bought it.
- You customized it (wallpaper, apps, ringtones).
- You carried it everywhere.

The network was still crucial, but now **the device** was the center of the experience.

The Wrong Assumption: "The Browser Is Enough"

The first instinct from many web-native companies:

> "People will just use our website on their phone browser."

They imagined:

- Pinch-zooming desktop layouts.
- Tiny tap targets.
- Horizontal designs squeezed into a vertical frame.

This is exactly the kind of mistake this Part IV is about:
treating the new era as a minor variant of the old one.

Smartphone reality said no.

The Actual Break: Apps Take Over

What actually happened:

- **Apps** became the primary unit of interaction:
  - Installed once, tapped often.
  - Got their own icon, notifications, background behavior.

- “Mobile-first” design became a thing:
  - Single-column layouts.
  - Big touch targets.
  - Short, focused flows.
- Browsers didn’t disappear, but they moved to the background.

The deep shift:

> The primary interface moved from “a website on a generic machine”
> to “an app on *my* device.”

That’s a clean 6 → 12 move: from diffused network space to concentrated personal object.

Who Got Caught Flat-Footed

Casualties of this break shared common traits:

- **Desktop arrogance**
  - “Our web experience is great; we don’t need a special mobile flow.”
- **Reluctance to rebuild**
  - “We’ll just ‘make it responsive’ and call it a day.”
- **Underestimating sensors and push**
  - Ignoring location, camera, background sync, notifications — all things phones were uniquely good at.

They saw the smartphone as “the web, but smaller” instead of “a new 12 p.m. object with its own logic.”

Who Surfed the Wave

Winners embraced:

- Apps as the default interface.
- Short sessions, more frequent touchpoints.
- Push notifications as a primary re-engagement lever.
- Deep use of sensors:
    - location-aware services,
    - camera-based experiences,
    - health and motion tracking.

They didn't just copy-paste their website.
They asked: "What does this thing *want* to be great at?"

The Lesson for AI → Robot

The AI → Robot transition is structurally similar:

- Web → Smartphone:
    - Network → Thing
    - Browser → Apps
- AI → Robot:
    - Network → Thing
    - Platform AI → Personal Robot

If you treat robots as "just another way to reach our chat bot," you're doing the "just use our website in mobile Chrome" move all over again.

History's hint:

> When a new 12 p.m. object shows up, you don't shrink the old UX;
> you design a new one around the object's constraints and powers.

Quick Exercise

Think of one well-known web-era product (news, social, banking, shopping).

1. How is its smartphone experience *actually* different from its old desktop website?
2. What did it gain by going app-first (notifications, offline, sensors)?
3. Now answer: what are the robot-era equivalents of those gains?

You're building intuition for where robots will change the rules, not just the screen size.

Chapter 21 — The Browser-to-App Moment (and Why Today's AI Bots Are Already Outdated)

This chapter is the hinge between history and now.

We're going to zoom in on one specific moment:

- when people assumed the **browser** on phones would be enough,
- and reality shifted to **apps** instead.

Then we'll map that directly to AI chatbots and robots.

1. What We Thought Would Happen

When smartphones landed, the comforting story was:

> "No need to rethink much. People will browse the web on their phones."

The assumptions:

- The browser is the universal interface.
- Websites just need minor tweaks to be usable.

- Existing UX patterns (menus, tabs, multi-column layouts) will survive.

Translated to AI today, this is:

> "We'll build a great AI chat bot on our website/app,
> and users will come talk to it."

Same pattern. Same comfort.

2. What Actually Happened: Apps Ate Mobile

Real users voted differently:

- Browsers stayed, but **apps** took the prime real estate.
- We got:
  - app stores,
  - home screens,
  - notifications,
  - deep linking,
  - in-app purchases.

The browser became:

- one of many apps,
- often a fallback, not the primary.

This is the break:
what used to be *the* interface became just another icon.

3. The Mapping: Today's AI Bots = Yesterday's Desktop Websites

Now: look at how companies are deploying AI today.

- On-site chat widgets: “Ask our AI assistant.”
- In-app bots: “Talk to our AI guide.”
- Branded copilots: “Our bot will help you navigate our platform.”

They assume:

- users will continue to:
  - come to their domain,
  - find their bot,
  - have a direct conversation inside their UX.

This is structurally identical to:

> “Users will just use our desktop website in their phone browser.”

In a robot world, that assumption dies.

4. The Coming Break: From “Their Bot” to “My Robot”

When robots become normal, the main interface will be:

- Human → **Their Robot**

The robot then:

- goes to Amazon,
- talks to the bank,
- deals with airlines,
- interacts with SaaS tools,
- fills out forms,
- calls APIs,

- manages errors.

In that world:

- The human does **not**:
  - go to your site,
  - open your app,
  - hunt for your chat bubble.

They simply say:

> "Can you handle this?"

and their robot:

> "handles this," by using **your tools**, if you've provided any.

Your AI bot is no longer the star.
It's the desktop-era website in a mobile world.

5. What Actually Matters in That World

In the browser-to-app transition, UX priorities shifted from:

- pixel-perfect websites → app performance, notifications, and flows.

In the AI→Robot transition, priorities shift from:

- clever bot UX → **clean capabilities and tools**.

Robots will judge you on:

- What can you actually do for my human?

- How reliable is your API surface?
- Are your policies machine-readable and predictable?
- Are your errors clean, or do you throw HTML and captchas at me?

Humans will judge you on:

- “Did my robot get it done or not?”
- “Do I ever have to step in and suffer a UI myself?”

6. The Lesson: Stop Optimizing the Soon-to-Be Edge

Just like desktop websites became the edge case for mobile life,
site-native AI bots will become the edge case in a robot world.

The center will be:

- the robot’s orchestration,
- the tools and connectors you provide,
- the contracts you honor.

You’re not trying to win by building the best “Our Bot.”
You’re trying to be:

> the easiest, cleanest, most capable node in **Their Robot**’s world.

Quick Exercise

Pick any current AI bot you know (your own product or another).

1. Describe how a human uses it today.
2. Now describe how a robot would ideally talk to the same company on behalf of that

human.

3. List what's missing today (tools, APIs, policies) to make #2 real.

That gap is the browser-to-app delta for AI → robots.

Chapter 22 — The Coming Break: From Platform AI to Personal Robots

Now we pivot fully into the next break:
from **Platform AI** (today) to **Personal Robots** (tomorrow's 12 p.m.).

You can think of this as:

- From "I go to *their* AI"
- To "I talk to *my* robot, which deals with their systems."

1. Where We Are: Platform AI at ~4 p.m.

Right now (2025-ish):

- Most serious AI usage is:
  - ChatGPT, Gemini, Claude, Grok, etc.
  - Embedded copilots inside platforms (Office, Figma, IDEs, CRMs).

Patterns:

- You *visit* the AI:
  - a website,
  - an app,
  - a chat window.
- You are a **user**, not an owner:

    - accounts, logins, subscriptions, usage tiers.

This is classic 6 p.m.-in-progress behavior.
AI is diffusing through the network — not yet embodied, not yet owned.

2. What “Robot Noon” Looks Like

Robot Noon is what happens when:

- Intelligence is **embodied** in things you own:
    - glasses,
    - home units,
    - mobile bots,
    - maybe humanoids in some contexts.
- These things:
    - know you deeply,
    - are persistent over years,
    - are the primary interface you talk to.

You don’t “go to AI” anymore. You live with it.

Examples of the experience:

- “Can you reorder what we got last month, but more kid-friendly?”
- “Can you find us a long weekend trip that doesn’t wreck the budget?”
- “Can you get this medical bill sorted out and put it in our records?”

You’re not thinking about platforms. You’re thinking about your robot’s competence and loyalty.

3. What Breaks for Platform AI

When this happens, several current assumptions crack.

Assumption 1: Users will keep coming to our domain.

- In Robot Noon:
  - traffic is mediated by robots,
  - humans rarely see your UX,
  - your “engagement” is API calls, not pageviews.

Assumption 2: We can own the conversation via Our Bot.

- In Robot Noon:
  - the primary conversation is Human ↔ Their Robot,
  - you’re in the background, talking Robot ↔ Platform.

Assumption 3: Personalization = “our model fine-tuned on your data.”

- In Robot Noon:
  - personalization lives primarily in the robot’s memory,
  - platforms see only the slices they need to fulfill a job.

Assumption 4: We can optimize for ourselves and call it For You.

- In Robot Noon:
  - if the robot perceives conflict between platform incentives and owner’s interests, it will route elsewhere.

The break is not just interface-level; it is about **who owns context and loyalty**.

4. The New Default: Single Personal Interface, Many Backends

Think structurally:

- Today:
  - many interfaces → many backends
  - you go to bank UI, airline UI, retailer UI, etc.
- Robot Noon:
  - one primary interface (your robot) → many backends
  - you talk to the robot, it talks to everyone else.

That means:

- Identity:
  - anchored in the robot (owner + robot pair),
  - delegated to backends as needed.
- Context:
  - primarily stored in the robot's memory,
  - only slices shared per-job.
- Trust:
  - sits with the robot,
  - platforms must earn the robot's repeated calls.

This is a deep inversion of today's platform AI logic, where platforms assume they own context and the user relationship.

5. What You Should Be Building for the Break

To survive (and thrive) through this break, the work is:

- Shift your roadmap from:
  - "Our bot," "Our assistant," "Our copilot,"
- To:

    - “Our tools,” “Our connectors,” “Our robot-native capabilities.”

Concretely:

- Expose the jobs you’re great at as clean tools:
    - order, refund, schedule, move money, verify, compare, etc.
- Make policies machine-readable:
    - fees, limits, timelines, eligibility, guarantees.
- Support robot-level accounts and permissions:
    - spend caps, domain restrictions, time bounds.
- Design for owner-first alignment:
    - no dark patterns that force robots into conflict with the humans they serve.

You’re trying to be the **best node in a robot’s capability graph**, not the place where people come chat.

6. Timeline Thinking: What Happens Before Full Noon

Before we reach full robot ubiquity, we’ll see:

- Wearables (AI glasses, pendants, pucks) that:
    - already behave more like 12 p.m. objects,
    - are bought, named, worn, and used as “mine.”
- Hybrid patterns:
    - people still use platform AI sometimes,
    - but increasingly offload recurring patterns to embodied devices.
- Early robot-native services:
    - companies treating robots as first-class customers,
    - building connectors before they build on-site bots.

If you wait for robots to be everywhere before you adapt, you are the desktop-only site in 2012.

Quick Exercise

Write a one-page “internal memo from the future” dated at full Robot Noon.

Answer:

- What percentage of your meaningful traffic comes from robots, not humans?
- What parts of your current AI strategy look obviously obsolete from that vantage point?
- What did you build that aged well because it was robot-native from the start?

You’re forcing yourself a few turns past the break, which tends to clarify what actually matters now.

Chapter 23 — Lessons from Missed Transitions: Who Didn’t Make the Jump and Why

Every break leaves casualties. This chapter is about the *patterns* behind those casualties, not dunking on specific logos.

We’ll use past transitions — PC → Web, Web → Smartphone — as mirrors to see what might go wrong in AI → Robot.

Pattern 1: Denial (“This Isn’t Really Different”)

The first and most common failure mode:

> “This new thing is basically the old thing with a twist.”

- PC → Web:

    - “The web is mostly for brochures, news, and email. Serious software will stay local.”
- Web → Smartphone:
    - “Mobile browsing will be the main use case. Apps are for games and toys.”

Companies in denial:

- Underinvest, late.
- Build superficial adaptations (a mobile landing page, a web syncing addon).
- Treat the new era as a fad that will fade back into the old normal.

This is the “it’s just a feature” mindset applied to something that’s actually a new substrate.

AI → Robot version of this:

- “Robots are just another form factor for our platform AI.”
- “We’ll ship a robot skin for our existing product and call it done.”

Pattern 2: Porting Instead of Rethinking

Second failure pattern:

> “We’ll just port what we already have, as literally as possible.”

- PC → Web:
    - People tried to mimic desktop windowing and menus inside browsers.
    - Heavy, clunky UI that ignored the web’s strengths (links, forms, statelessness).
- Web → Smartphone:
    - Giant desktop pages stuffed into a tiny viewport.
    - No thought for one-handed use, spotty connectivity, short sessions.

Porting is comforting because:

- It preserves mental models.
- It reuses existing code and assets.
- It gives the illusion of “we’re already there.”

Problem:

- It almost never makes use of the **new medium’s strengths**.
- It carries forward the **old medium’s constraints**.

AI → Robot version:

- Giving your existing platform AI a physical shell, but:
  - still thinking in terms of “sessions,”
  - still assuming the user comes to you,
  - still hoarding identity and context on the platform side.

Pattern 3: Cannibalization Fear

Some players *see* the change but fear it:

> “If we go all-in on the new thing, we’ll hurt our existing business.”

- PC → Web:
  - Software vendors afraid that SaaS would undercut license revenue.
- Web → Smartphone:
  - Web-first companies afraid that apps would fragment their analytics and loosen control.

So they:

- Ship half-hearted versions.
- Hide the new option.
- Price it awkwardly.
- Let internal politics slow everything down.

AI → Robot version:

- Platforms afraid that:
    - exposing tools and connectors will reduce on-site usage,
    - robots mediating interactions will kill direct engagement metrics,
    - letting robots own context will weaken their data moats.

It's not that they don't understand the future. It's that they're structurally incentivized to drag their feet.

Pattern 4: Wrong Metrics, Wrong Winners

Metrics shape behavior. During transitions, they can mislead you.

- PC → Web:
    - "Installs sold" vs "active users."
    - Companies optimized around license revenue even as usage shifted online.
- Web → Smartphone:
    - "Pageviews" vs "in-app actions, retention, daily opens."
    - Some teams declared mobile "not that important" because mobile pageviews were low — while native app usage exploded.

AI → Robot version:

- Measuring:
    - prompt volume,

    - chat session length,
    - user time-in-bot.

When the real future metrics should be:

- task completion rate via robot connectors,
- robot-originated volume,
- “friction-free” jobs done with zero human UI.

If you worship the wrong numbers, you’ll protect the wrong things.

Pattern 5: Culture That Can’t Let Go of Its Own Importance

Some organizations simply can’t handle being demoted from “primary interface” to “backend provider.”

- Web → Smartphone:
    - Teams deeply attached to their pixel-perfect desktop UIs
    - had trouble seeing that, on mobile, their logo might be just one tile in a grid.

The companies that navigated it best:

- Detached ego from interface ownership.
- Were willing to become pipes, platforms, or components if necessary.

AI → Robot version:

- Companies emotionally invested in:
    - “our assistant,”
    - “our copilot,”
    - “our conversational brand voice,”

may struggle to accept:

- that they're more valuable as clean tools in someone else's orchestration,
- that **Their Robot** is the real interface people care about.

The Question for You

These patterns aren't historical trivia. They're diagnostic tools.

For your own work, ask:

- Where are we minimizing the robot shift as "just another channel"?
- Where are we porting old UX instead of inventing robot-native flows?
- What are we afraid to cannibalize?
- Which metrics would have to change if robots truly were our primary customers?
- Where is our ego tied up in owning "the conversation" instead of owning the competence?

Your honest answers tell you how likely you are to become part of the "who didn't make the jump" chapter of the next book.

Quick Exercise

Write a one-page "postmortem from 2035" about a fictional (or real) company that failed the AI → Robot transition.

Answer:

- What did they see coming but fail to act on?
- Which of the patterns above did they fall into?
- What did their successful competitor do differently?

Then flip it: underline every sentence that feels uncomfortably close to decisions you see around you today.

Chapter 24 — Pattern Library: How Breakpoints Usually Announce Themselves

Transitions rarely arrive as press releases. They leak in through side doors.

This chapter is a **pattern library** for early signals — the little glitches in the Matrix that usually show up before the hand on the clock fully swings to the next position.

You can use it as a checklist for spotting the shift from Platform AI to Robots in real time.

1. Pattern: "Toy" Use Cases That Don't Go Away

New substrates often show up first as "toys":

- Early web: personal homepages, forums, fan sites.
- Early smartphone: games, novelty apps, simple messaging.

Experts dismiss them:

- "Cute, but not serious."
- "Fun for teens, not for real work."

But if those toys continue to grow, mutate, and pull attention, they're often:

> the **safest place** to explore the new substrate,
> where constraints and affordances get discovered.

Robot-era version:

- Robot pets, simple household devices, playful embodied toys.
- Wearables that feel more like “gadgets” than serious tools.

If you see those use cases compounding instead of fading, that’s a tell.

2. Pattern: Users Start Hacking Around the Old UX

Before transitions are acknowledged, users start doing weird things:

- PC → Web:
  - Emailing themselves files to move between machines.
  - Using webmail instead of local clients.
- Web → Smartphone:
  - Taking photos of screens as reminders.
  - Using messaging apps as to-do lists or file stores.

These hacks say:

> “The old UX isn’t covering the real job anymore.”

Robot-era version:

- People rigging up voice assistants, shortcuts, scripts, and browser automations to simulate “a thing that just does it for me.”
- Using phones as quasi-robots:
  - voice notes to self,
  - auto-forwarding emails to bots,
  - gluing services together with Zapier/IFTTT-style flows.

When hacks pile up, the substrate is waiting to be formalized.

3. Pattern: Edge Populations Adopt First

Breaks rarely start with “everyone.”

Instead, you see:

- Professionals with extreme needs (traders, designers, engineers).
- People with constraints (disabilities, high travel, heavy logistics).
- Subcultures that adopt new tools aggressively.

PC → Web:

- Developers, sysadmins, academics were early heavy web users.

Web → Smartphone:

- Young users, commuters, field workers leaned into mobile first.

Robot-era version:

- People with:
  - chronic time scarcity,
  - caregiving responsibilities,
  - mobility or sensory constraints,
  - high coordination load (family logistics, complex work).

If robots (or proto-robots) start becoming non-negotiable for these edge groups, the center is next.

4. Pattern: Metrics Go Sideways for the Old Interface

A classic early warning is metrics that flatten or decline in the old interface while overall demand still rises.

- Web → Smartphone:
    - Desktop pageviews peak, then stagnate, even as overall Internet use increases.
    - Mobile-native metrics quietly explode somewhere else.

Robot-era version:

- Time-in-bot or prompt count flattens, but total "AI-mediated tasks" continue to rise.
- On-site bot usage stagnates while API calls from agents and integrators grow.

If you see value moving but your dashboard doesn't, it means:

> "The real shift is happening somewhere you're not measuring yet."

5. Pattern: Developers Get Bored with the Old Thing

Another tell: where the **interesting work** goes.

- PC → Web:
    - The best young devs stopped building Windows shareware and started building web apps.
- Web → Smartphone:
    - Hackathons, meetups, excitement all moved to iOS/Android.

Robot-era version:

- Builders:
    - more excited about agent frameworks, connectors, robotics SDKs, tool ecosystems
    - less excited about "one more chat UI with a different gradient."

When the talent and curiosity move, the rest of the economy eventually follows.

6. Pattern: Regulation Starts to Lag and Then Snap

At breakpoints, regulation looks confused:

- PC → Web:
  - Laws designed for physical distribution struggled with digital copying.
- Web → Smartphone:
  - Privacy, tracking, app store rules arrived late and messy.

Robot-era version:

- Questions about:
  - liability when robots act,
  - data ownership between robots, platforms, and humans,
  - safety, employment, and autonomy.

When regulators suddenly pay attention to something that “wasn’t real yet,” it’s often because the transition is already well underway.

7. Pattern: Language Shifts

Language gives you early signals:

- PC era:
  - “software,” “programs,” “disks.”
- Web era:
  - “sites,” “URLs,” “online,” “dot-com.”
- Smartphone era:
  - “apps,” “push,” “swipe,” “mobile-first.”

Robot-era language will sound like:

- “my robot,”
- “my assistant,”
- “my glasses,”
- “talk to my agent,”
- “have it handle that for you.”

When non-technical people start talking this way without irony, you’re close to our 12 p.m.

8. Using the Pattern Library

This library isn’t meant to be exhaustive; it’s meant to tune your attention.

For the AI → Robot transition, regularly ask:

- Where are the “toys” that refuse to die?
- What hacks are people doing to get a robot-like experience from non-robot tools?
- Which edge groups are adopting embodied AI first?
- Which of our metrics are suspiciously flat despite apparent demand?
- Where are the most ambitious builders aiming their energy?
- What new legal and cultural arguments are starting that didn’t exist five years ago?
- How are normal people describing all this in their own words?

Breakpoints almost always whisper before they shout.

Quick Exercise

Build your own “breakpoint radar”:

1. Pick three signals from the pattern library that you think will be most important in your domain (e.g., retail, finance, healthcare).

2. For each, write:
    - what it would look like in practice,
    - how you'd measure or observe it,
    - what you'd change in your roadmap if you saw it clearly.

Treat this as a living document. As the robot era approaches, you'll be updating it often.