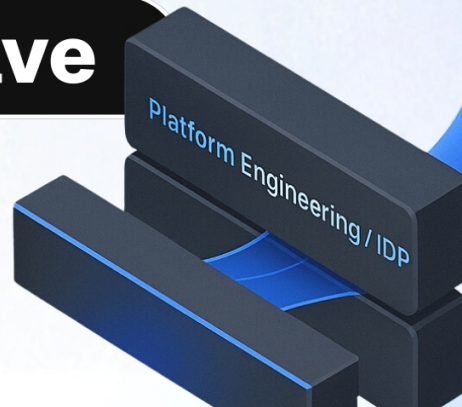


Surviving the Agentic AI Wave

Platform Engineering for
Telecom & Healthcare



Agentic AI in Telecom & Healthcare: The Platform Engineering Playbook

Posted on December 16, 2025 by Sony Battina

Telecom networks are under constant pressure: exploding data, 5G rollouts, OTT competition, and customers expecting zero downtime. Healthcare is juggling digital front doors, virtual care, electronic health records, and strict regulations all while clinical and operations teams are stretched thin. Now comes the **Agentic AI wave**; autonomous AI agents that don't just answer questions but act, opening tickets, updating records, triggering workflows, even making recommendations in real time.

Used well, Agentic AI can transform both telecom and healthcare and here's why:

- Faster network troubleshooting
- Smarter triage and scheduling
- Proactive customer and patient outreach
- Automated claims and billing validation

Used poorly, it can create:

- Integration chaos
- Compliance nightmares
- Runaway cloud and AI costs

The real question for telecom and healthcare leaders is no longer: “Should we adopt agentic AI?”

It’s: “Do we have the platform to survive and scale it?” That’s where platform engineering and engineering partners like [TechTez](#) come in.

What Is Platform Engineering?

Platform engineering is the practice of building an **internal product** often called an Internal Developer Platform (IDP) that your own teams (and now AI agents) use to build, deploy, and run software safely and consistently.

Instead of each team reinventing how to:

- Connect to OSS/BSS or EHR systems
- Deploy services to Kubernetes or cloud
- Set up observability, security, and compliance checks

They use a standardized platform that internal platform team designs.

Think of it as:

- A self-service layer for developers, data teams, and AI agents
- A catalog of reusable building blocks: APIs, data products, workflows
- A set of guardrails baked into the platform: identity, access, logging, encryption, approvals

Learn more about IDP with [google cloud](#)

What This Looks Like in Practice

Platform engineering doesn't replace DevOps or SRE, it amplifies them by turning your environment into a product that teams can safely and reliably use.

- A telecom engineer spins up a new microservice or AI agent using a template that already includes logging, metrics, security, and network configuration.
- A healthcare product team connects an AI triage assistant to patient data via a standard FHIR-based API provided by the platform without touching the EHR directly.
- An AI-powered packet analyzer or claims validation engine plugs into the same platform, using governed, auditable access to core systems.

Why “Random AI Projects” Cost More Than You Think



Random **AI Projects** Cost More Than You Think



In both telecom and healthcare, it's tempting to spin up fast AI POCs:

- A chatbot for customers
- A triage assistant for patients
- An “experimental” network troubleshooting agent

They look cheap and quick. But without a platform, the hidden costs pile up.

1. Fragmented Integrations: Each POC integrates differently: direct DB access here, legacy SOAP call there, screen scraping somewhere else. Nothing is reusable and every new project starts from zero.

2. Compliance Blind Spots: In healthcare, PHI flows through notebooks and custom scripts. In telecom, subscriber data is accessed without proper masking, logging, or role-based access.

3. Runaway Costs: Multiple teams hit LLMs and AI services directly, with no central cost governance, caching, or routing strategies. Cloud and AI bills spike before value is realized.

4. Operational Risk: Agents can trigger actions (network changes, appointment updates, claims decisions) without standardized approval of flows or rollback mechanisms. A “small POC” can create very real incidents.

5. Slowed Innovation: Each AI initiative spends months redoing the basics of security reviews, access requests, and connectivity because there's no shared platform foundation. In mission-critical industries, this isn't just inefficient; it's dangerous.

5 Ways Platform Engineering Helps Telecom & Healthcare Survive the Agentic AI Wave

1. A Safe Control Plane for AI Agents

Instead of letting agents talk directly to core systems, the platform provides well-defined, policy-driven APIs and tools. For example, when we design agentic workflows:

- Agents don't connect directly to EHRs or BSS systems.

- They call platform-managed services that:
- Mask sensitive data by default
- Enforce identity and access policies
- Log every interaction for audits
- Apply rate limits and anomaly detection

Result: You get an AI that can act not just chat without handing it unrestricted access to your most sensitive systems.

Read more about how TechTez designs [Agentic Workflows](#)

2. Standardized Data Access Across Silos

Both sectors suffer from data fragmentation:

- **Telecom:** OSS/BSS, CRM, mediation, network probes, NMS
- **Healthcare:** EHR, PACS/RIS, LIS, CRM, claims, wearables

Platform engineering turns this into governed data products and consistent access layers:

- APIs to fetch “subscriber context” or “patient context” in a single call
- Feature stores feeding multiple models (churn prediction, readmission risk, fraud scores)
- Read-only, masked views designed specifically for AI and LLMs

Build RAG or agentic systems on top of these, that aren’t hacking around bespoke integrations; they’re consuming clean, governed data through the platform.

3. Compliance & Security by Design

Healthcare has HIPAA/GDPR. Telecom has lawful intercept, data retention, and privacy obligations. Compliance can’t be an afterthought.

A well-designed platform makes compliance default:

- Centralized identity and access management (IAM)
- Data classification, tokenization, and masking policies
- Encryption in transit and at rest standardized across services
- Built-in audit trails for both human and AI actions

Design “golden paths” like:

- Pre-approved architecture blueprints for AI workloads
- Standard ways for agents to query and update systems
- Policy-as-code libraries that every service must use

So instead of re-reviewing every AI initiative from scratch, you can say:

“If it’s built on the platform and follows this golden path, it’s already aligned with 80–90% of our compliance needs.”

4. Faster Delivery, Lower Cognitive Load

Platform engineering gives teams self-service capabilities:

- Spin up new microservices, data pipelines, or AI tools using ready-made templates
- Use shared observability, tracing, and error-handling patterns out of the box
- Register APIs and data products into a central catalog so agents know what they can call

Examples from a TechTez Project:

- A telecom team launches a new **AI-powered Network Packet Analyzer** as a service on the platform decoding PCAP files, highlighting SIP/RTP anomalies, and raising incidents through a standard ticketing interface.
- A healthcare team deploys an **AI-assisted claim validation engine** that cross-checks diagnostic codes, procedure codes, and policy rules using a governed set of APIs and a central rules engine, no custom plumbing each time.

Teams can focus on domain problems instead of infrastructure and glue code.

5. Observability & FinOps for AI Workloads

When AI hits production, leaders need to see:

- Which agents are calling which systems
- How much specific workflows cost (tokens, compute, storage)
- Where failures, bottlenecks, and compliance risks exist

A good platform brings:

- **End-to-end traces:** “User → Agent → Orchestrator → Microservices → Databases”
- **Unified dashboards:** Network performance, clinical workflows, and AI agent actions in one place
- **FinOps controls:** Tagging, cost allocation, rate limits, and auto-optimization for AI calls

Learn more about [CNCF projects](#) about Platform engineering in FinOps

This is critical when you’re mixing:

- High-volume telecom traffic
- Sensitive patient data
- Expensive AI inference costs

Platform engineering makes AI costs **visible, explainable, and controllable**.

Tangible Benefits of a Platform-First AI Strategy

Benefits of a Platform-First AI Strategy



Adopting platform

engineering for agentic AI delivers:

- **Resilience:** Standardized rollout and rollback patterns, fewer surprises
- **Speed:** Launch new AI workflows in weeks, not quarters
- **Compliance:** By-design logging, masking, and access control
- **Cost Control:** Shared AI infrastructure, caching, routing, and FinOps visibility
- **Happier Teams:** Engineers and clinicians focus on value, not plumbing

How Telecom & Healthcare Can Get Started:

You don't need a "perfect platform" on Day 1. Start small, think product, and iterate.

Step 1: Map Pain Points & Critical Journeys

Work with teams to understand:

- Where time is being lost today
- Where integration is fragile or risky
- Which systems EHR, PACS, OSS/BSS, CRM, network elements are central

Step 2: Pick 2-3 "Hero Use Cases"

Choose a mix of high-value and feasible use cases, such as:

- **Telecom:**
 - AI-assisted network ticket triage and auto-remediation
 - Packet-level anomaly detection using the AI-powered Network Packet Analyzer
- **Healthcare:**
 - AI-enabled patient intake and appointment routing
 - AI-driven claim validation and coding support

These use cases define what the first version of the platform must support: APIs, data access, orchestration, observability, and policies.

Step 3: Define the Platform MLP (Minimum Lovable Platform)

Instead of boiling the ocean, define a compact, valuable first version:

- A simple self-service portal or CLI
- Service and agent templates (logging, security, observability included)
- API gateway and IAM integration

- A governed connection to a small but critical set of data sources
- Basic FinOps and monitoring for AI calls

Step 4: Pilot, Measure & Iterate

Launch your first AI + platform-powered journeys into a controlled production environment.

Track:

- Time to first deploy for new services/agents
- Time to get data access (compared to legacy approach)
- Incident frequency and rollback effort
- AI and infra costs per workflow

Use these learnings to refine platform templates, guardrails, and self-service capabilities.

Step 5: Treat the Platform as a Product, Not a Project

Platform engineering works when you:

- Have a dedicated platform team (internal)
- Maintain a roadmap driven by real user (internal team) feedback
- Publish documentation, patterns, and training
- Measure platform success via developer experience, incident reduction, and time-to-market

In telecom and healthcare, this platform quickly becomes a strategic asset and the foundation on which all future AI sits.

Measuring ROI: Platform Engineering in the Agentic AI Era

ROI shows up in several dimensions.

1. Direct Cost Savings

- Less duplicated tooling and infrastructure
- Fewer custom integrations to maintain
- Reduced manual effort in deployment, monitoring, and compliance checks

2. Faster Innovation

- Shorter cycle from idea → prototype → production
- Ability to launch and test more AI-powered experiences without months of groundwork

3. Reduced Risk & Compliance Exposure

- Lower chance of outages from inconsistent deployments
- Better control over PHI/subscriber data access and flows
- Stronger audit trails for regulators and internal risk teams

4. Talent Productivity & Retention

- Engineers and data scientists work on meaningful problems, not repetitive setup
- New team members onboard faster via templates and documented patterns

5. Better Patient & Subscriber Experience

- Proactive issue detection in telecom networks
- Faster triage, scheduling, and claims handling in healthcare
- Consistent, AI-augmented experiences across channels

A simple way to start quantifying: **ROI (%) = (Annual Gains - Platform Costs) ÷ Platform Costs × 100** Where **Annual Gains** include:

- Engineering hours saved
- Reduced incidents and downtime
- Faster launch of new AI services
- Avoided compliance penalties and rework

Final Thoughts: Build the Platform Before the Wave Gets Bigger

Telecom and healthcare don't get do-overs when systems fail. The agentic AI wave is already here helping teams:

- Diagnose network issues faster
- Assist clinicians and operations with complex workflows
- Automate claims, support, and back-office tasks

Without a platform, this wave can:

- Create integration chaos
- Increase risk and cost
- Slow you down instead of speeding you up

With a strong platform, supported by partner, the same wave becomes your **competitive advantage**. The best time to invest in platform engineering was a few years ago.

The second-best time is **today**.

FAQ's:

1. Why is platform engineering critical for AI in telecom and healthcare?

Because both industries are complex, regulated, and mission-critical. Platform engineering provides a safe, standardized foundation for AI and agentic workflows to run without compromising reliability or compliance.

2. How does TechTez fit into this picture?

TechTez partners with enterprises to design and build internal platforms, AI-powered tools (like [packet analyzers](#) and claim validation engines), and the guardrails needed to safely run agentic AI at scale.

3. Can platform engineering help control AI and cloud costs?

Yes. By centralizing how models are accessed, logged, and monitored, the platform enables caching, routing, right-sizing of models, and clear cost allocation key for FinOps.

4. What should we build first?

Start with an MLP: templates, API gateway, IAM integration, observability, and governed access to 1-2 critical data domains. Then expand as teams adopt the platform.

5. Is this only for large operators and hospital chains?

No. Smaller telecom providers, digital health startups, and mid-size hospitals benefit even more, they can deliver enterprise-grade capabilities with lean teams by leaning on a strong platform and partners like TechTez.

6. How do we measure ROI from a platform-first Agentic AI strategy?

Track the before/after on time-to-market, integration effort, incident reduction, compliance rework, and AI/cloud spend per workflow. ROI typically comes from reusable templates, governed access, fewer outages, and FinOps controls.

$$\text{ROI (\%)} = (\text{Annual gains} - \text{Platform costs}) \div \text{Platform costs} \times 100.$$