Architecting Systems to Architecting Agility

Recurring Patterns Part 2

Part 4 of 7: Efficiency Communication & Treadmill

Pattern 3 The Efficiency Gap

Speed Cost Sensitivity Needs

- We weren't always as **fast or cost-sensitive** as the business needed.
- Example Connected vehicle/IoT data explosion.
- Default response often scaling horizontally (Hadoop Spark).

Efficiency Gap Big Data Costs

Investment Skills vs Alternatives

- While powerful these required **significant infrastructure investment** specialized skills.
- Felt more focus needed on **efficient data modeling** partitioning lifecycle mgmt.
- Could potentially achieve goals at

fraction of cost/complexity.

Delivering insights faster.

Efficiency Gap AI Training Costs

Training From Scratch Dilemma

⚙ Witnessed worrisome trend teams globally training **large AI models from scratch**.

⚙ Using hundreds/thousands GPUs incredibly **expensive time-consuming**.

Efficiency Gap Lean AI Practices

Faster Cheaper Alternatives Ignored

⚙ Often cheaper faster alternatives existed.

⚙ **Fine-tuning** pre-trained models using **RAG/CAG**.

⚙ Could yield satisfactory results quickly economically.

⚙ Best practices for **lean AI**

development not always prioritized.

Pattern 4 Communication Silos

The Challenge of Unheard Voices

- In large complex programs (autonomous driving global IoT)...
- Ensuring critical **technical feedback reached decision-makers** at right time was constant challenge.

Communication Silos Lost Insights

Technical Debt Maintenance Costs

- Recall architects/engineers raising concerns about accumulating **technical debt**.
- Pointing out long-term **maintenance costs** of chosen stacks.
- Concerns sometimes **deprioritized or addressed too late** due to deadlines

silos.

Communication Silos Consequences

Predictable Problems Friction

- Not malice just the **friction inherent** in large complex systems.
- Led to predictable scaling problems performance bottlenecks costly refactoring.
- Valuable perspectives lost.

Pattern 5 The Tool Treadmill Intro

Solving Problems Creating Complexity

⚙ Industry evolution felt like solving one problem...

⚙ Only to introduce **another layer of complexity** to master.

Tool Treadmill Example Microservices

Autonomy vs Distributed Complexity

⚙ Monoliths to microservices (essential for SOTA scaling).

⚙ Offered team autonomy independent scaling.

⚙ But introduced **distributed systems complexities** (service mesh tracing sagas K8s).

Tool Treadmill The Energy Drain

Keeping the Machinery Running

⚙ Each step solved limitation but demanded **significant investment** learning managing new tooling.

⚙ Spent considerable energy just **keeping the machinery running**.

Lesson

Foundation For Change

- Recognizing these patterns complexity requirements efficiency communication tool complexity...
- Laid the groundwork for seeking a different approach.

Series Index

Part 1: The Pivot

Access Part 1 PDF

Part 2: The Technologist's Vantage Point

Access Part 2 PDF

Part 3: Pattern 1 Complexity Requirements

Access Part 3 PDF

Part 4: Pattern 2 Efficiency Communication Tool Treadmill (Current)

Access Part 4 PDF

Part 5: Vision AI Catalyst Plant Manager Example

Access Part 5 PDF

Part 6: AI Transforming the SDLC

Access Part 6 PDF

Part 7: Architecting Agility The Mission

Access Part 7 PDF

Read the Full Article: From Architecting Systems to Architecting Agility...

All resources mentioned are available at **https://agilp.org/pdf/**

Read the Full Article on LinkedIn

Connect & Engage

LinkedIn: https://www.linkedin.com/in/amitabhrjha/

X (Twitter): https://x.com/amitabhrjha

Web: www.agilp.org

Disclaimer & Acknowledgments

The opinions expressed are my own & don't necessarily represent my employer's views. My perspective is constantly evolving, shaped by invaluable interactions with friends, colleagues, mentors, insightful authors, and industry influencers - thank you all! Much of this content, including these carousels, is co-created with AI co-pilots like ChatGPT, Gemini, and Grok. My intent is to synthesize knowledge and share it back with the community.