alok.giri@ncit.edu.np

Why Study Case Studies?

- Theoretical Agile ≠ Real-World Agile
- Case studies help us:
    - Understand **what works in practice**
    - Learn from **mistakes and challenges**
    - Apply patterns to our own teams

SUCCESS CASE – SPOTIFY

Slide 3: Spotify – Overview

- Founded in **2006**, now a global streaming leader
  - Faced challenge: How to **scale Agile** without losing **speed and autonomy**

Spotify's Motto:
"Think it. Build it. Ship it. Tweak it."

Spotify Agile Model

Spotify didn't **invent Agile**, but evolved a unique **structure** to scale it:

1. Squads

- Basic unit (~6–12 people), similar to a **Scrum team**
- Cross-functional (Dev, QA, UX, etc.)

- Fully autonomous: owns a feature or product area
- Uses Scrum or Kanban (team decides)
- Has a **Product Owner**

> *“A squad is like a mini-startup, responsible for a specific aspect of the product.”*

Spotify – Tribes, Chapters, and Guilds

2. Tribes

- A **collection of squads** (up to ~100 people)
- Squads in a tribe work on related areas (e.g., playlist, mobile)
- Led by a **Tribe Leader**
- Promotes alignment and cross-squad learning

3. Chapters

- Horizontal group of people with similar skills (e.g., Frontend Devs)
- Belong to **different squads** but the **same tribe**
- Led by a **Chapter Lead** (like a line manager)
- Focus: Tech consistency, skill growth

Guilds

- Informal communities across **tribes and squads** ● Focused on shared interest or practices (e.g., testing, DevOps, design)
- Anyone can join; helps with **knowledge sharing**

FAILURE CASE – NOKIA

Nokia's Agile Attempt

- Dominant phone brand in early 2000s
- Introduced Agile (Scrum) ~2007–2009
- **Intent:** Respond faster to smartphone disruption

- **Reality:** Agile implementation failed to save them

FAILURE CASE – AIRBUS A350

Slide 9: Airbus A350 Agile Integration

- Applied Agile in developing cockpit software
- Teams were **distributed across Europe**
- Some teams followed Agile; others used **Waterfall** or traditional systems engineering

Why did Nokia Fail?

Root Cause	Details
Agile in name only	Adopted ceremonies, not mindset
No cultural change	Hierarchical management resisted team autonomy
Delivery ≠ Product Fit	Agile speed didn't help bad strategic decisions (e.g., Symbian)
Lack of feedback loops	Market signals ignored; no real product feedback
Blame culture	Fear stifled innovation and ownership

“We had Scrum, but not agility.”

Why was Airbus slow?

Issue	Consequence
Lack of systemic agility	Agile in isolation = misaligned deliveries
Delayed integration	Found mismatches in late-stage testing
Geographical and cultural silos	Poor collaboration and communication
No full value stream ownership	Agile teams couldn't see impact on full sys

Result: Costly delays, rework, and partial Agile rollback

PATTERNS AND INSIGHTS

Patterns of Agile Success

1. Teams own what they build
2. Cultural fit with Agile mindset
3. Fast feedback and experimentation
4. Strong leadership support and trust
5. Continuous improvement baked in

Patterns of Agile Failure

1. “Agile theater” – rituals without real change
2. No stakeholder alignment
3. Blame and control-based culture
4. Tech debt ignored
5. Teams constrained by top-down decisions

The Cultural Foundation

Positive Culture	Toxic Culture
Autonomy & trust	Micromanagem
Psychological safety	Blame-shamin
Learning mindset	Fear of mistak
Transparency & collaboration	Siloed thinking

Spotify's success was cultural before structural.

Continuous Feedback Loops

Why Feedback is Crucial:

- Enables **fast correction**
- Drives **product relevance**
- Empowers teams to **reflect and improve**

Feedback in Agile Layers:

- **Code level:** Pull requests, pair programming
- **Team level:** Retrospectives
- **Customer level:** Reviews, usability testing
- **System level:** Product analytics

What is Agile Transformation?

Definition:

Agile Transformation is the **fundamental change** in how an organization **thinks, delivers value, collaborates, and operates**, by adopting **Agile principles and frameworks**.

It's not just about:

- Doing daily stand-ups
- Installing Jira
- Running sprints

It **is about:**

- **Mindset shift**
- Structural change
- New ways of **leading, working, and learning**

Why Organizations Choose Agile Transformation

- Faster response to market changes

- Higher customer satisfaction
- Improved team morale and engagement
- Increased visibility and predictability
- Better alignment between business and IT

Key Steps in Agile Transformation

Step 1 – Create Urgency & Executive Buy-in

- Make a strong **business case for change**
- Highlight:
    - Customer dissatisfaction
    - Innovation lag
    - Delivery bottlenecks
- **Involve leadership** early
- Secure executive sponsors to champion the change

Quote:

"Without leadership support, Agile becomes theater."

Step 2 – Define the Vision and Agile Goals

● What does **Agile success** look like for this organization? ● Define outcomes: Faster releases, better quality, more engaged teams

Step 3- Assess Current State (As-Is Analysis)

- Evaluate:
  - Team structures
  - Delivery processes (e.g., Waterfall, ad hoc)
  - Culture and decision-making
  - Tooling and technical practices

Tools:

- Agile Maturity Assessment
- Value Stream Mapping

- Surveys and interviews

Step 4- Design the Target State

- Choose transformation models:
  - **Team-level:** Scrum, Kanban
  - **Scaling frameworks:** SAFe, LeSS, Spotify, Nexus

Design considerations:

- Agile team structures
- Role realignment (PO, SM, dev teams)
- Governance and metrics

Step 5- Form Agile Teams & Define Roles

- Form **cross-functional teams** with clear ownership
- Assign:
  - **Product Owner**

- **Scrum Master or Agile Coach**
- Developers, Testers, Designers

Tip: Avoid partial teams or role confusion. Full-time commitment yields better agility. **Step 6- Start with Pilots (Proof of Concept)**

- Select **2–3 pilot teams**
- Choose manageable scope
- Give them training, tools, and support

Why Pilots?

- Quick wins
- Learn what works (and doesn't)
- Build momentum with real outcomes

Step 7 – Scale What Works

- Use successful pilots to define **playbooks**

Roll out across more teams and business units ●
Support with:
    ○ Agile Coaches
    ○ Communities of Practice
    ○ On-demand training

Scaling Options:

- SAFe for structured orgs
- Spotify model for innovative orgs
- LeSS for product-centric orgs

Step 8 – Measure, Inspect, and Adapt

- Track transformation KPIs:
    ○ Cycle Time, Lead Time, Team Velocity
    ○ Employee Engagement

    - Customer Feedback
- Conduct **transformation retrospectives**
- Adjust based on feedback and metrics

Continuous transformation > One-time switch

From	Form of Resistance
Senior Management	Fear of losing control
Middle Management	Threat to authority
Teams	Comfort with known processes
Support Functions	"This doesn't apply to us" mindset

Challenge 2 – Misunderstanding Agile

Symptoms:

- Teams “doing Scrum” but not being Agile

- Overfocus on tools and ceremonies
- Agile becomes “delivery speed only” game

Solution:

- Emphasize **Agile values & principles**
- Prioritize **mindset over mechanics**

Challenge 3 – Lack of Role Clarity

- Confusion between **Project Manager** and **Scrum Master** ●
POs expected to write specs instead of manage product vision ●
SMs becoming status reporters instead of coaches

Fixes:- Clear role definitions, Role-based training, Leadership modeling correct behavior

Challenge 4 – Silos and Poor Collaboration

- Agile thrives on **collaboration**

- Silos block:
  - Shared ownership
  - Flow of value
  - Feedback loops

Antidotes:

- Cross-functional teams
- Shared KPIs
- Frequent cross-team planning (e.g., PI Planning)

Challenge 5 – Inconsistent Leadership Behavior

- Saying “Be Agile” while demanding fixed scope/deadlines
- Micromanagement or ignoring Agile values

Solution:

- Train and coach leaders
- Use **Agile Leadership frameworks** (e.g., SAFe LPM, Lean Thinking)

Team Formation and Role Assignment

Agile Planning – Sprint 0

Activities in Sprint 0:

- Understand the **project objective**
- Break down **features into user stories**
- Prioritize the backlog (use MoSCoW or story mapping)
- Estimate effort (e.g., story points)
- Create the **Sprint Plan** (1-week or 2-week iterations)

Tools: Trello, Jira, Notion, physical boards (if in-person)

Iterative Development and Delivery

During Each Sprint:

- **Daily Standups:** Brief updates on progress, blockers
- **Task board updates:** Move cards across Kanban states
- **Sprint Review:** Demo to stakeholders/instructor
- **Sprint Retrospective:** Reflect on process and teamwork

Deliverable Format:

- Working prototype, UI mockup, MVP demo, or service simulation

Continuous Feedback and Adjustments

Internal Feedback:

- Use retrospectives to improve collaboration and delivery
- Adjust WIP, sprint size, or roles as needed

External Feedback:

- Instructor or stakeholder reviews during Sprint Reviews
- Respond with story re-prioritization or change planning

Reminder: Agile is adaptive—not rigid

Preparing the Final Presentation

Presentation Structure (10–15 mins):

1. **Project Summary:** What problem did you solve? 2. **Team Roles & Dynamics:** Who did what? How did Agile work for you?

3. **Planning Process:** Tools used, how backlog was created
4. **Sprint Demonstrations:** Snapshots or walkthroughs of deliverables
5. **Retrospective Learnings:** What went well? What changed?
6. **Next Steps:** If you had more time, what would you improve?

Format: PowerPoint, live demo, Miro board, or recorded walkthrough

Peer Review and Feedback

Each team member rates peers (anonymous):

- Contribution to sprint tasks
- Collaboration and communication
- Accountability and team spirit

Scoring Format (1–5):

- 1 = Rarely contributed
- 5 = Consistently exceeded expectations

Peer scores can influence **final grades or feedback coaching.**