From Binary Decisions to Infinite Possibilities

Management in the AI + Quantum Era

How We Used to Make Decisions

"Should we do
A or B?"

"Let me think
for 2 weeks..."

"Option B.
Final answer."

"Hope it works!"

This worked when the world was slow and predictable.
It doesn't work anymore.

How Decisions Work Now With AI:

"Here are 12 options with trade-offs"

"Path 3 has 73% success rate"

"Go with 3, keep 7 as backup"

"Adjust in real-time as data comes in"

Not "yes or no" → "what's most likely to work, and what's plan B?"

A Real Example From My Work

The Problem:

I track 100+ engineering changes across multiple product releases for several global teams. Each change has a status, an owner, settings data, and deadlines.

The Old Way:

Open each item. Read it. Copy data into a spreadsheet. Email managers. Wait. Repeat next week. Takes DAYS.

The AI Way:

I describe what I need in plain English: "Scan all open items, find stuck ones, build a dashboard, flag what needs attention." Done in MINUTES.

What "Using AI" Actually Looks Like

👤 Me:

"I have 130 items across 5 teams. Find the ones stuck >14 days. Group by manager."

🤖 AI:

"I'll fetch status & dates, filter for >14 days, group by owner, generate a live dashboard. Want SLA % too?"

👤 Me:

"Yes. Make it refresh automatically every day."

🤖 AI:

"Done. It validates data before publishing and keeps old version if anything fails — never blank."

Case Study: When Things Go Wrong (And AI Helps)

Script crashes.
Dashboard blank.

Managers see
empty page.

AI: "Write to temp
first, validate,
then publish"

System now
self-protects.

The Lesson

AI doesn't just do tasks — it helps you think about what could go wrong.
It's a thinking partner that catches edge cases you'd miss at 5pm on a Friday.

Key: Build systems that fail gracefully. AI helps you think of scenarios you'd miss.

5 AI Skills Every Manager Needs

- **1. Break problems into steps** — "First get data, then filter, then visualize" not just "make a report"
- **2. Verify AI's work** — Always check: does this make sense? Trust but verify.
- **3. Iterate quickly** — First version won't be perfect. Say "good, now also add X" and improve.
- **4. Spot what's automatable** — If you do it the same way every time → AI can do it.
- **5. Save and reuse** — Don't solve the same problem twice. Build once, use forever.

Quantum Computing

Normal Computer

Tries paths ONE at a time.

Like solving a maze by trying every turn yourself.

Quantum Computer

Tries ALL paths at once.

Like having a million copies of yourself, each trying a different route.

Why managers care: Some problems (scheduling, logistics, resource planning) have millions of answers. Quantum evaluates them ALL simultaneously.

How YOU Will Use Quantum:

You won't write quantum code. You'll ASK quantum-powered tools questions:

You'll say:

"Given these 40 engineers, these 15 projects, their skills, preferences,
and upcoming PTO — what's the best team assignment for next quarter?"

You'll say:

"We have 6 releases, 200 features, 30 dependencies between them.
Show me the shipping order that gets the most value to customers fastest."

You'll say:

"Our build takes 4 hours. Which 20% of tests catch 99% of real bugs
for THIS specific code change? Skip the rest."

The interface will feel exactly like AI does today — you ask in English, you get an answer.

The only difference: the answers will be OPTIMAL, not just "pretty good."

Case Study: Smarter Testing & Bug Triage

❌ The Old Way: Run Everything or Guess

A product has 80,000 regression tests. Full run = 6 hours.
Every code change triggers a full run. Engineers wait. Releases slip.
Manual triage: "skip the ones that probably don't matter" — but bugs leak through.

🤖 AI Today: Smart Selection (Good)

AI analyzes code change → maps affected paths → picks 8,000 relevant tests.
Run time drops from 6 hours to 45 minutes. Catches 96% of bugs.

⚛️ Quantum Future: Optimal Selection (Perfect)

Quantum evaluates ALL possible test combinations simultaneously.
Finds the MINIMUM set that guarantees 99.9% coverage for THIS exact change.
Run time: 20 minutes. Bug escape rate: near zero.

Manager impact: Ship faster, with MORE confidence, using FEWER resources.

When Will Quantum Actually Reach Software Managers?

📍 2024-2026: AI Is Your Daily Co-pilot (NOW)

AI writes code, AI summarizes meetings, AI builds dashboards.
You're already using the first wave. Quantum comes INSIDE these tools.

📍 2027-2028: Quantum Powers Your Existing Tools

JIRA / project tools start offering "quantum-optimized sprints."
CI/CD pipelines use quantum for test selection.
You won't know it's quantum — it'll just be faster & better.

📍 2029-2030: Quantum Becomes Invisible Infrastructure

Resource planning, architecture decisions, release sequencing —
all handled by quantum-AI hybrid systems. Manager asks a question,
gets the mathematically best answer with trade-offs explained.

- **Don't wait for quantum to "arrive"** — build AI skills NOW (same interface, same muscle)
- Identify problems where "good enough" costs you money (scheduling, allocation, testing)
- When quantum-powered features appear in your tools — you'll be first to benefit

Why AI + Quantum Together = Game Changer

🤖 AI Gives You

- Understands your words
- Spots patterns in data
- Generates new ideas
- Automates workflows
- Learns from past results

⚛️ Quantum Adds

- Explores ALL options at once
- Solves "impossible" puzzles
- Simulates complex systems
- Finds the TRUE best answer
- Handles million-variable problems

Together: AI understands your problem → Quantum finds optimal answer → AI explains it in plain English.

5 Things to Remember

- 1. AI is a multiplier, not a replacement.
- 2. You don't need to code. You need to think clearly and ask good questions.
- 3. Start small, iterate fast. First try won't be perfect — that's fine.
- 4. Quantum is coming (2-3 years). It solves the 'impossible' problems. AI handles everything else today.
- 5. Winners won't be the smartest. They'll be the most curious and adaptable.

Stop Choosing A or B.
Start Exploring A through Z.

The future doesn't belong to people who know all the answers.
It belongs to people who ask the right questions.

Thank You!