

# The Entrepreneurial Pendulum: From Intrapreneurship to Independence and Back

Dr. Asankhaya Sharma





# The Career Journey

Act 1: Intrapreneurship Phase (Senior Software Engineer at Microsoft)

Act 2: First Entrepreneurship Experience (Head of R&D at SourceClear)

Act 3: Intrapreneurship at Scale (Engineering Director at Veracode)

Act 4: Entrepreneur (Co-Founder at Patched.Codes)



# X-trapreneurship

**This Session:** The complete journey between both worlds

**My Core Message:** You don't have to choose one path forever

Each transition makes you stronger

The pendulum swing is your competitive advantage



# Starting as an Intrapreneur at Microsoft

I joined Microsoft right after my undergrad university.

My initial role was matched with the internal SAP team because I had done an internship with SAP.

The role was very boring ...



# Rotational Early Career Program

As my first act of rebellion, I left SAP team to join the Microsoft APEX (Accelerated Professional Experiences Program)

Many companies offer such early career rotational programs (that typically place you in different roles in the company for a short duration)



# The Reality of Corporate Innovation

APEX program allowed me to work in different roles -  
Software Engineer, Program Manager, Research Engineer  
etc. across the company - MSIT, MSR.

Advantages of a large company:

- Resources, stability, and scale
- But also had committees, approvals, and politics



# Innovation Opportunities



At the largest private hackathon on the planet, Microsoft employees fire up ideas by the thousands



## TAKE A THINK WEEK

Twice a year, Bill Gates goes somewhere secluded to have a THINK WEEK. All communications with his family, friends, and employees are banned. He brings a stack of books and uses this time to read, come up with new ideas, and work on personal development. A lot of break-throughs at Microsoft stemmed from these "Think Weeks."

IG-MOTIVETHOUGHT

MOTIVETHOUGHT.ORG

## The Garage is a program that drives a culture of innovation

We deliver programs and experiences to our employees, customers, and ecosystem that drive collaboration, creativity, and experimentation.

## Hacking at Microsoft

Opportunities for employees, interns, and customers to leverage their creativity and encourage collaboration





# What I learned

- How enterprise customers think
- What slows down innovation
- Why big companies struggle with speed
- How to build your network

## **Why Ex-Intrapreneurs Make Better Entrepreneurs:**

We know what enterprises actually need (not what they say)

We understand long sales cycles

We can build for scale from day one

We've seen what doesn't work



# First Taste of Entrepreneurship

In 2010, I left Microsoft to do my PhD at NUS

In 2014, I joined my ex-manager from Microsoft to work at a startup SourceClear

2014-2018, Built the SourceClear R&D center in Singapore and grew the team from 0 to 40



# Building SourceClear in Singapore

## **Singapore Advantage:**

- Government support (grants, programs)
- Access to Asia and West
- Lower burn rate than Silicon Valley
- Talented engineers from local universities

## **Key Decisions:**

- Focused on enterprise from day one
- Built security into DevOps workflow



VENTURECAPITAL BLOG

# SourceClear Raises \$10M to Secure Open-Source Code

By Deborah Gage

Updated Oct. 27, 2015 1:21 pm ET

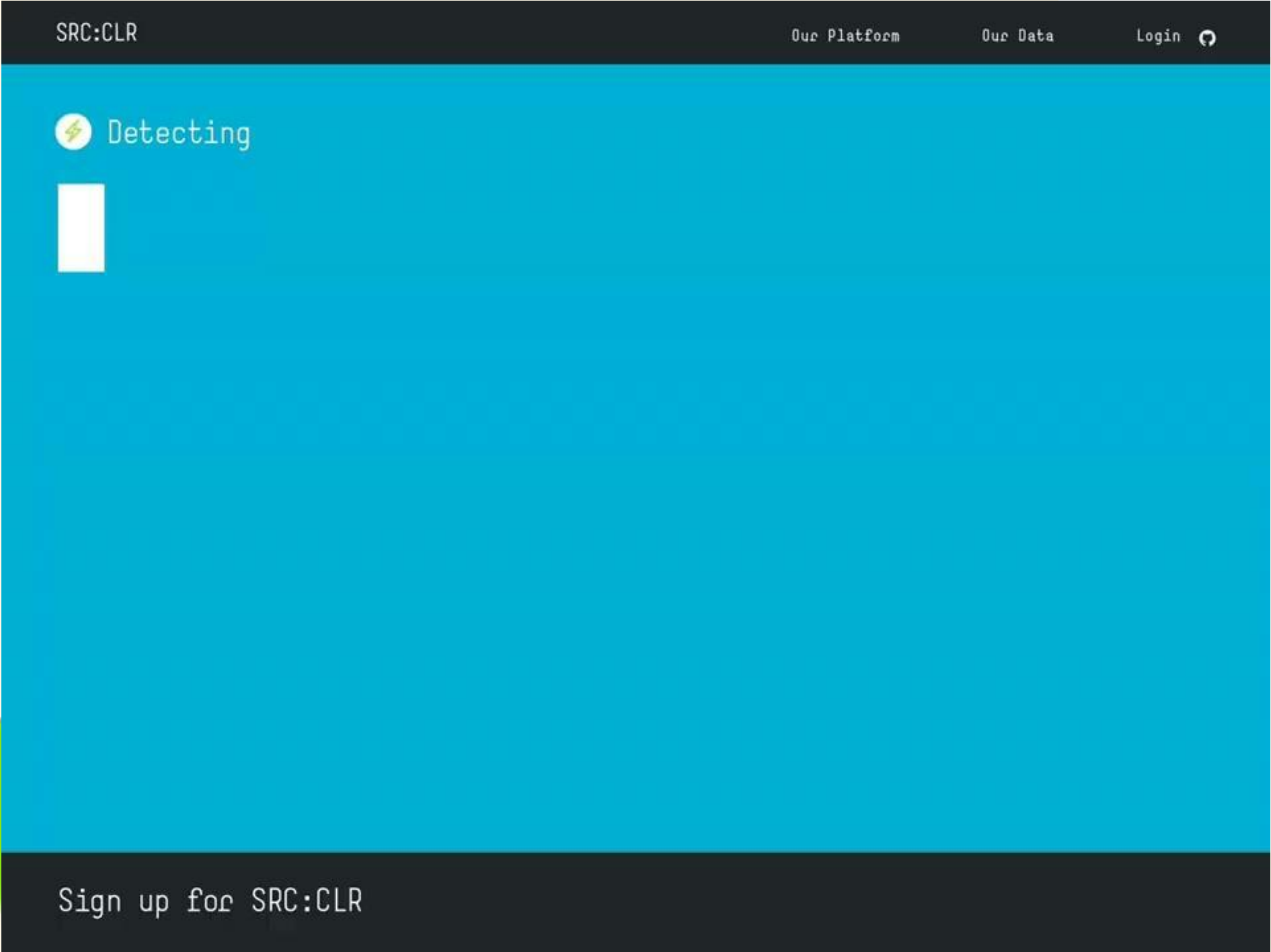


Share



Resize

Mark Curphey worked to stamp out software bugs for about a decade as head of the security tools team at [Microsoft](#) Corp. and in several other jobs before he realized that the problem was getting worse instead of better.



## CA Technologies acquires SourceClear for its DevSecOps portfolio

Latest News Published: April 12th, 2018 - Ian C. Schafer

CA Technologies announced its acquisition of software composition analysis specialists SourceClear early this week with aims to incorporate SourceClear's SaaS-based SCA tool and proprietary vulnerability database with their Veracode cloud platform.

"We are excited about what this acquisition means for our customers in terms of increased support for SCA in DevSecOps environments and the ability to confidently use open source components without introducing unnecessary risk," Sam King, general manager of CA Veracode wrote in a [blog post](#)





# **Intrapreneurship 2.0**

## **With Founder DNA**

How to Be an Effective Intrapreneur After Being a Founder:

- You understand urgency in a way career employees don't
- You can bridge startup innovation with corporate execution
- You can spot opportunity where others see process



# Leading Innovation at Veracode

Implemented some of the best practices from my experiences at Microsoft and SourceClear:

- A new Incubation team
- Two hackathons a year
- Data Science and Machine Learning guild
- New products - Container Security and Auto Remediation
- Acquired companies - Hunter2 and Jaroonna



# Why I Left Again - The AI Revolution

2022: The Pattern Repeats

- GPT-3 showed AI's potential
- Saw opportunities the company wouldn't pursue
- Hit the innovation ceiling again
- Had conviction about timing (AI wave)
- Felt the itch to build, not just lead

The Realization:

- Intrapreneurship has limits
- Some waves require entrepreneurship
- Corporate innovation can't match startup speed for emerging tech



# The Second Swing - Patched.Codes


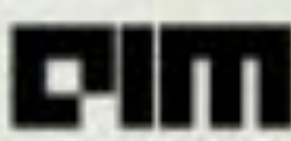
What's Different This Time:

- Know exactly who will buy (enterprise security teams)
- Can navigate funding/acquisition from day one
- Have network from both corporate and startup worlds
- Can pattern-match 10x faster

**Y Combinator**

Make something people  
want.





This YC-Backed Startup Automates Code Reviews and Documentation for Enterprises

10,000+ Monthly Downloads

# Patch the gaps in Enterprise IT & Ops.

Agentic workflows that cut MTTR and AHT, boost FCR, and keep your SLA—without changing your tools.

[Book a demo](#)[View Runbooks](#)

```
rohan@LAPTOP-IETGENSP: ~/j x + v
rohan@LAPTOP-IETGENSP:~/sample-injection$ patchwork AutoFix --config ../patchwork-con
figs/patchflows

EXPLORER
SAMPLE-INJECTION [WSL: Ubuntu]
index.py
README.md

index.py
1 import sqlite3
2
3 CONFIG = {
4     "default_table": "users",
5     "default_column": "username"
6 }
7
8 def get_data_by_config_value(value):
9     query = "SELECT * FROM " + CONFIG["default_table"] + " WHERE " + CONF
10
11     connection = sqlite3.connect("database.db")
12     cursor = connection.cursor()
13     cursor.execute(query)
14     result = cursor.fetchall()
15     connection.close()
16
17     return result
18
19 # Test
20 print(get_data_by_config_value("admin"))
21
```





# The Data Behind The Pendulum

- 70% of unicorn founders had significant corporate experience
- 45 is the average age of successful startup founders
- 50% of YC founders have corporate backgrounds
- 20% time at Google created Gmail, AdSense, Google News



# Patterns form my own Journey

- Corporate experience → Domain expertise
- First startup → Execution skills
- Back to corporate → Scale thinking
- Second startup → Everything combined



# **Your career is 40+ years**

## **Plan accordingly**

Years 1-3: Learn the Industry (Intrapreneur)

- Join company with real innovation culture
- Find mentors who've done both paths
- Document frustrations (future startup ideas)

Years 4-7: Build or Join Startup (Entrepreneur)

- Apply domain knowledge from corporate
- Move fast, break things, learn quickly
- Build network of other founders



# Career Roadmap

Years 8-12: Scale Your Impact (Choice)

- Return to corporate with founder credibility
- Start second company with more wisdom
- Join venture capital or accelerators

Years 13+: Innovation Arbitrage

- Move based on opportunity, not identity
- Leverage accumulated advantages
- Mentor the next generation



# Singapore's Unique Ecosystem

## Government Support for BOTH Paths:

For Intrapreneurs:

- Innovation grants for corporations
- IMDA innovation programs
- GovTech opportunities

For Entrepreneurs:

- Startup SG grants
- Block71 ecosystem
- Enterprise Singapore support

## Why This Matters:

- You can swing between both easily
- Government supports transitions
- Network effects in small ecosystem



# The Decision Framework

## Which Path Should You Take First?

Some suggestions based on my own experience



# Choose Intrapreneurship First If:

You don't know the industry well

You need to build savings

You want to learn on someone else's dime

You value mentorship and structure



# Choose Entrepreneurship First If:

You have a burning problem to solve

You have 18+ months runway

You can handle extreme uncertainty

You have relevant domain expertise



# The Third Option - Hybrid Models:

Venture studios (structured entrepreneurship)

Corporate accelerators (entrepreneurial intrapreneurship)

Consulting (exposure to many models)

Part-time founding while employed



# The Meta Lesson

You Are Not Your Current Role

You're not "an intrapreneur" or "an entrepreneur"

You're an innovator who chooses appropriate vehicles

Each role is a chapter, not your entire story



# First Principles Thinking

The Framework We'll Use:

- Identify Core Problem - What are we really trying to solve?
- List Current Assumptions - What does everyone believe?
- Challenge Each Assumption - Ask "Is this actually true?"
- Find Fundamental Truths - What can't be changed?
- Rebuild from First Principles - Create new solution
- Apply Context - Adjust for corporate vs startup



# The Open Source Security Problem

Every company uses open-source libraries. The average application has 200+ dependencies. Yet nobody knows what vulnerabilities are hiding in their code. Major breaches happen weekly. Let's solve this problem.



# STEP 1: IDENTIFY CORE PROBLEM

**Surface Level:** Companies keep getting hacked through open-source vulnerabilities

Q: "Why are they getting hacked?"

A: "They don't know what vulnerabilities exist in their code"

Q: "Why don't they know?"

A: "They can't see into their dependencies"

**CORE PROBLEM:** "Organizations are blind to risks in their software supply chain"



## **STEP 2: LIST CURRENT ASSUMPTIONS (What Everyone Believes)**

"We need to scan all source code for vulnerabilities"

"Security scanning must be comprehensive to be valuable"

"Developers won't change their workflow for security"

"We need our own vulnerability database"

"More scanning = better security"

"Security tools must run separately from development"

"Only security teams care about vulnerabilities"

"If it's open-source, someone else is checking it"

"Vulnerability scanning takes time because it needs to be thorough"

"Companies want complete security reports"



# STEP 3: CHALLENGE EACH ASSUMPTION

Assumption 1: "We need to scan all source code"

Challenge: Do vulnerabilities come from YOUR code or dependencies?

Reality: 90% of vulnerabilities are in dependencies, not custom code

Status: ✗ FALSE

Assumption 2: "Security scanning must be comprehensive"

Challenge: Do you need to find ALL issues or CRITICAL ones?

Reality: 80/20 rule - 20% of vulnerabilities = 80% of risk

Status: ✗ FALSE



Assumption 3: "Developers won't change workflow"

Challenge: What if security was invisible to them?

Reality: They will if it doesn't slow them down

Status: ✗ FALSE

Assumption 4: "We need our own vulnerability database"

Challenge: Where does vulnerability info actually live?

Reality: It's in public commits, CVE databases, GitHub issues

Status: ✗ FALSE

Assumption 5: "More scanning = better security"

Challenge: Does volume equal value?

Reality: Better to fix 10 critical than find 1000 minor

Status: ✗ FALSE



# STEP 4: FIND FUNDAMENTAL TRUTHS

After challenging everything, what remains true?

Fundamental Truths:

- ✓ Dependencies introduce risk (can't be changed)
- ✓ Vulnerabilities exist in public information (factual)
- ✓ Speed matters - hackers don't wait (time sensitivity)
- ✓ Developers control what gets into production (workflow reality)
- ✓ Companies need to know their risk level (business requirement)

These are bedrock truths we build upon.



# STEP 5: REBUILD FROM FIRST PRINCIPLES

The Logic Chain:

If: Dependencies introduce risk (Truth 1)

And: Vulnerability info is public (Truth 2)

And: Speed matters (Truth 3)

And: Developers control deployment (Truth 4)

And: Companies need risk visibility (Truth 5)

Then: Solution must:

Monitor dependencies, not all code

Pull from public sources in real-time

Integrate into developer workflow

Show risk level instantly

Focus on what matters most

First Principles Solution: "Real-time dependency monitoring that shows critical risks during development"



# STEP 6A: APPLY CONTEXT - INTRAPRENEURIAL SOLUTION

"Security Risk Dashboard Initiative"

Frame as Risk Reduction (not new tool)

Pilot Approach - Start with one willing team 3-month  
proof of concept

Security team: "Reduces your workload"

Dev team: "No workflow changes"

Legal: "Compliance automated"

Finance: "ROI in 6 months"



# STEP 6B: APPLY CONTEXT - ENTREPRENEURIAL SOLUTION

"SourceClear - Security for Developers"

CLI tool only Scans package.json/pom.xml

Returns top 10 risks Free for open-source

Talk to 100 developers in month 1

Find the hair-on-fire problem

Iterate based on feedback

Bottom-up adoption

Developers use free, Companies pay for private repos



# Thank You!

LinkedIn → <https://www.linkedin.com/in/asankhaya/>

Twitter → <https://x.com/asankhaya>

Email → asankhaya at yahoo dot com

How to be an authentic Leader? →

<https://www.amazon.com/dp/B0CKYT5RKQ>

Questions?