Senior Program Manager Product team Tracxn Case Study - Pradeep Dhayanandan

Executive Summary

This document outlines a comprehensive program strategy for building an AI powered system to extract and maintain accurate address information from company websites. The program will process over 100,000 companies monthly while maintaining precision standards critical to Tracxn's data operations.

Problem Statement

Tracxn needs a production system that can automatically discover, extract, validate, and maintain current address and pincode data from company websites at scale. This is not just about web scraping but it's about building a sustainable, scalable system that handles both scheduled batch updates and a real time request.

What makes it particularly challenging is the diversity we will be dealing with. Company websites come in all shapes and sizes that is from simple one pager to complex multinational sites. Addresses might be hiding in footers, contact pages, PDFs, or even images and dealing with international formats like US zip codes, UK postcodes, Indian pincodes each have their own quirks.

Why this matters? Bad address data isn't just an inconvenience it affects customer trust, breaks downstream analytics, and can cost us deals. When a sales team is on a call and needs accurate company location data, every second counts.

Prerequisites & Ground Rules

What we need to get right from day one?

Technical Realities

- **Website issues:** Every company have unique pattern. We need to handle everything right from modern React applications to the legacy HTML from 2005.
- **The address issues:** Sometimes it's nicely structured in JSON-LD, sometimes it's scattered across three different pages, and sometimes it's in a PDF download.
- **Global complexity:** What works for extracting Silicon Valley addresses might fail spectacularly for companies in Bangalore or Berlin.

Business Boundaries

- **Budget:** API costs for large language models rise fast we must use them selectively, not for every extraction.

- **Quality:** One wrong address in a client report can damage relationships that took years to build.
- **Speed expectations:** users call this API during meetings so the responses must feel instantaneous. Even five seconds feels like forever during urgency.
- **Team Readiness:** Before we jump in designing the program we need to understand:
    - Have our teams worked together on something this complex before?
    - What infrastructure can we leverage vs what needs building?
    - Who has bandwidth, and who's already stretched thin?

The Program Design

Step 1: Building the Workflow & Ownership

Phased Approach

MVP Phase (Weeks 1-6): Prove It Works

- **Scope:** 1,000 handpicked companies we know well
- **Goal:** Can we actually extract addresses accurately?
- **Team:** Keeping it lean 2 engineers, 1 ML engineer, 1 QA
- **Success looks like:** 95% accuracy on this controlled set

Pilot Phase (Weeks 7-12): Testing the Limits

- **Scope:** 10,000 companies across regions like US, India, and Europe
- **Goal:** Does it scale? What breaks? How much does it cost?
- **Team expansion:** Adding 2 Data Ops folks for validation
- **Success looks like:** Maintaining quality while finding cost optimizations

Production Scale (Week 13+): Full Throttle

- **Scope:** 100K+ monthly batch, 1K+ daily on-demand
- **Goal:** Autonomous operation with minimal hand holding
- **Full team:** Everyone's in, with on call rotations
- **Success looks like:** It just works, and does not require much intervention

Who Owns What? (Ensuring there is no finger pointing in the future when things go south)

Database Integration

Infra Team's Responsibility includes:

- Deduplication of records
- Schema management and versioning of extracted addresses
- Ensuring smooth integration into production databases
- Logging and audit trails for traceability
- Infra team will also ensure monitoring of data freshness and rollback mechanisms if any faulty data enters production

Discovery & Extraction Pipeline

Data Engineering Team's Turf:

- Building and maintaining the web crawlers
- Finding the right pages contact, about, locations, whatever works
- Storing everything properly so we can trace our steps
- Making sure we don't get blocked by crawler blocking websites

AI/ML Team's Domain:

- Crafting prompts that actually extract addresses, not random text
- Deciding when to use expensive models vs. cheaper alternatives
- Building confidence scores that we can actually trust
- Learning from mistakes to get better over time

The handoff: Engineering gives clean HTML → ML returns address with confidence scores. Simple, clear, no ambiguity.

Processing & Normalization

This is where it gets collaborative:

- **Data Ops:** Defines what "good" looks like and handles the weird edge cases
- **Engineering:** Turns those rules into code, validates against external sources
- **Product:** Decides what formats our customers actually need

Critical decision: If confidence drops below 80%, it goes to human review.

Conflict Resolution (When Sources Disagree)

Automated checks handle:

- Obvious duplicates
- Format validation (like is this even a valid pincode?)
- Cross referencing with Google Places or postal APIs

Humans handle:

- Multiple office locations (like which is the HQ?)
- Conflicting sources (website says X, LinkedIn says Y)
- The ambiguous scenarios where it requires human intervention

Step 2: Measuring the metrics:

Precision - We target >95% precision to ensure that the extracted addresses are accurate, since even a single incorrect address can mislead customers and damage trust.

Recall - We target >90% recall to ensure comprehensive coverage of company addresses, since missing locations can be as damaging as incorrect ones. While 100% recall is impractical due to diverse website formats, maintaining above 90% ensures near complete coverage while balancing cost and scalability.

Metric	**Target**	**Target Audience**	**How We Track**
Precision	>95%	Product & Customers	Daily sampling, weekly reviews
Recall	>90%	Product & Customers	Validated against ground truth samples, periodic audits
Cost per million records	<$50k	Finance & Leadership	Real-time dashboard, bi-weekly optimization

Batch SLA	72 hours for 100K	Operations	Automated monitoring, escalation if delayed
API Response	<5 seconds (95th percentile)	Engineering & Sales	Real-time monitoring, PagerDuty alerts

Making Numbers Actionable:

Raw metrics are not beneficial without context. So, we can utilize it by

- **Precision drops below 95% or Recall drops below 90%?** Immediate alert to team lead/POC, finding root cause analysis for the issue.
- **Costs trending up?** Weekly optimization sprint to find savings
- **SLA breach?** Post mortem within 48 hours, fixes deployed within a week

Every metric has an owner, a dashboard, and an escalation path. No numbers should be meaningless.

Step 3: Governance

Meeting Rhythm

Weekly Standup

Who: Respective Team leads/ POCs
What: Blockers, resource needs, this week's priorities
Output: Clear action items are discussed

Monthly Business Review

Who: VP Product, Director of Engineering, Finance, myself(SPM)
What: Metrics deep dive, budget check, strategic pivots
Output: Executive dashboard, decision log

Quarterly Strategy Session

Who: C level stakeholders
What: ROI analysis, future investment, competitive positioning
Output: Next quarter's priorities, budget approval

Communication

- **Slack:** Day to day coordination, quick questions

- **Weekly email update:** Progress, metrics, upcoming changes etc
- **Confluence documentation:** The source of truth for everything

When Things Change (mostly they will)

For technical changes: Propose → Impact assessment → Rollout plan → Execute → Review

For business changes: Document requirements → Cross team review → Phased rollout → Measure impact

All changes are properly documented and communicated.

Risk Management

Risk: LLM APIs go down or hit rate limits

Impact: Processing stops, SLAs breached
Mitigation: Multiple providers on standby, fallback to regex for simple cases, smart queue management
Owner: ML Team with Infrastructure support

Risk: Websites start blocking us

Impact: Data gaps, manual intervention needed
Mitigation: Respectful crawling (we're guests, not invaders), rotating user agents, manual backup process
Owner: Data Engineering

Risk: Manual review becomes a bottleneck

Impact: Delays, quality issues, team burnout
Mitigation: 24/7 coverage across time zones, clear SLAs for review, continuous automation improvements
Owner: Data Operations

Risk: Costs spiral out of control

Impact: Budget overrun, program at risk
Mitigation: Daily cost monitoring, automatic throttling at 80% of daily budget, monthly optimization sprints
Owner: Program Manager

Defining Success

What Success Looks Like

At 6 Months:

- Processed 600,000+ companies successfully
- Maintained >95% precision and >90% Recall throughout
- Reduced manual touch from 30% to under 10%
- Cost per record below $40
- Zero critical SLA breaches

The Bigger Picture:

- Built a framework we can reuse for other extraction needs
- Established Tracxn as having the most accurate company location data
- Created a competitive moat through superior data quality
- Developed a team that can tackle the next big challenge

Why this approach will work?

what makes this program design different?

1. We start small and learn fast. No six months planning cycles that produce beautiful documents but no working code. We'll have real addresses extracted in week 2.

2. Everyone knows their roles/responsibilities. Clear ownership leads to less ambiguity, no finger pointing, and no "I thought you were handling that" moments.

3. Metrics that matter to real people. Not vanity metrics for PowerPoints, but numbers that directly impact our customers and our bottom line.

4. Built for sustainability. This isn't a project that we launch and forget. It's a living system with feedback loops, continuous improvement, and clear governance.

5. Honest about risks. Every program has risks. The difference is whether we acknowledge them upfront or discover them in production at 2 AM in the night.

Role of the Senior Program Manager

As the Senior Program Manager, my responsibility is to orchestrate delivery across cross functional teams. I ensure that roles, ownership, and interfaces are clearly defined. I track dependencies, risks, and budgets, and maintain alignment with business requirements. Importantly, my focus is not with directly writing crawlers, regexes, or prompts. Instead, I facilitate governance, communication, and accountability to ensure teams deliver outcomes aligned with Tracxn's goals.

Final Insights

Building this address extraction system is really about building organizational muscle. When we deliver this, we are not just getting better data but we are proving we can tackle complex, cross functional challenges and deliver real business value.

The teams involved will learn to work together, trust each other, and build something that matters. The infrastructure we create becomes the foundation for the next innovation. And the confidence we gain from succeeding here carries forward to bigger challenges.

That's what great program management enables, not just delivering a project, but elevating the entire organization's capability to execute.