

Agile Product Management



Course purpose: connect customer discovery with iterative product delivery



Move teams from shipping output to moving outcomes



Who it's for: product managers, owners, founders, delivery leads



Core promise: evidence-based decisions via discovery, experiments, and metrics



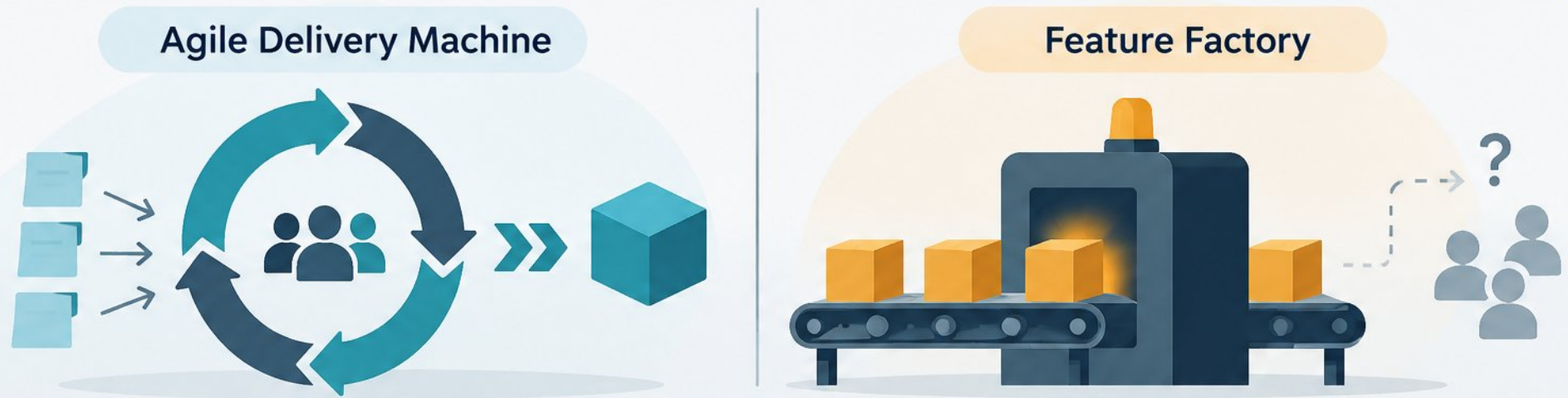
Journey: foundations, operating model, discovery, roadmaps, scaling, action plan



Apply immediately: pick one outcome metric and one risky assumption



Why Agile Delivery Alone Does Not Create Value



Agile delivery is how you build; agile product management is what you build



Teams can run perfect sprints and CI/CD while building the wrong product



Feature factory: output celebrated, adoption and behavior change never measured



Proxy backlogs separate strategy from delivery, so outcome accountability disappears



Empowerment in language only, without decision rights, becomes facilitation



If your roadmap is features with dates, you are a project organization

The Product Operating Model in 2026



Operating model: how structure, funding, governance, metrics and culture work together



Empowered teams are durable and cross-functional, assigned outcomes, not features



Five interlocking elements: culture, strategy, teams, discovery, delivery



AI makes building cheap, so judgment and decision quality become the constraint



Change all dimensions together; structure alone creates friction and failure



Roles, Decision Rights, and Team Topologies



Product manager

owns the why and what:
market, strategy, roadmap,
outcomes

Strategic direction



Product owner

is a Scrum accountability:
backlog, sprints, value
delivery

Execution accountability



Product trio

product management,
design, and engineering
discover together

Shared discovery



Product manager owns the why and what: market, strategy, roadmap, outcomes



Product owner is a Scrum accountability: backlog, sprints, value delivery



Product trio: product management, design, and engineering discover together



Decision rights: clear authority, escalation, and accountability prevent conflicts



Diagnostic: who can kill a started feature? If it needs a meeting, split



Scaling: one PO per team, one PM per one to three products

Decision
rights

Customer Discovery and Opportunity Framing



Discovery is continuous risk reduction, not a one-time project phase



Opportunity solution tree:
outcome → opportunities → solutions → assumption tests



Opportunities are unmet customer needs, pains, and desires in the customer's words



Ask about specific recent behavior, not hypothetical opinions



Minimum bar: one customer touchpoint per week by the building team



Keep the tree alive: update weekly, prune falsified branches, avoid a static poster



From Opportunities to Prioritized Bets



Backlog is a decision system, not a storage closet



Prioritize on learning value, reversibility, and customer impact



When building gets cheap, effort stops being the constraint



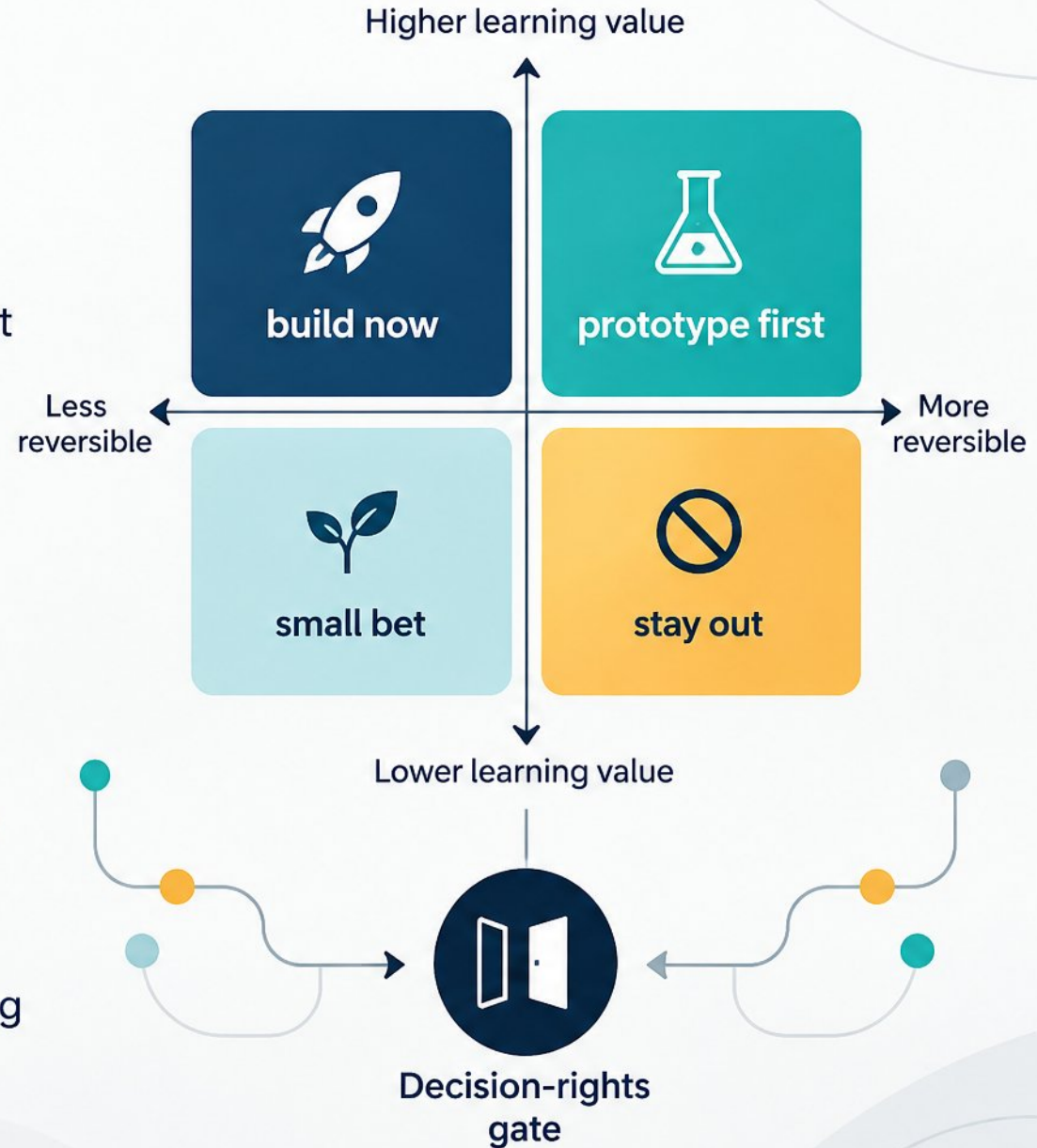
RICE, WSJF, and cost of delay must be adapted



Four boxes: build now, prototype first, small bet, stay out



Without real trade-offs, everything stays and focus disappears



Outcome-Based Roadmaps and Story Slicing



Replace feature roadmaps with outcomes: "Increase trial-to-paid conversion from 15% to 25%"



Now / Next / Later signals priority without false date precision



Trace every roadmap item to a key result: who does what, by how much, by when



Features are hypotheses for outcomes, not commitments



Story mapping and vertical slicing deliver value early, keeping slices small enough to learn from



One product outcome per timeframe; multiple outcomes dilute focus



Experiments, Hypotheses, and Iterative Delivery



Hypothesis: change, expected effect, metric, mechanism, failure conclusion



Match method to risk: demand risk → fake doors; value risk → prototypes, concierge



Pre-commit metric, MDE, duration, and decision rule before launch



Guardrail metrics catch local wins borrowed from elsewhere



Inconclusive is valid; resist shipping because you invested



Feed validated learning straight into the sprint backlog

Experiment Protocol



Hypothesis



Primary Metric



Guardrail Metric



Decision Rule



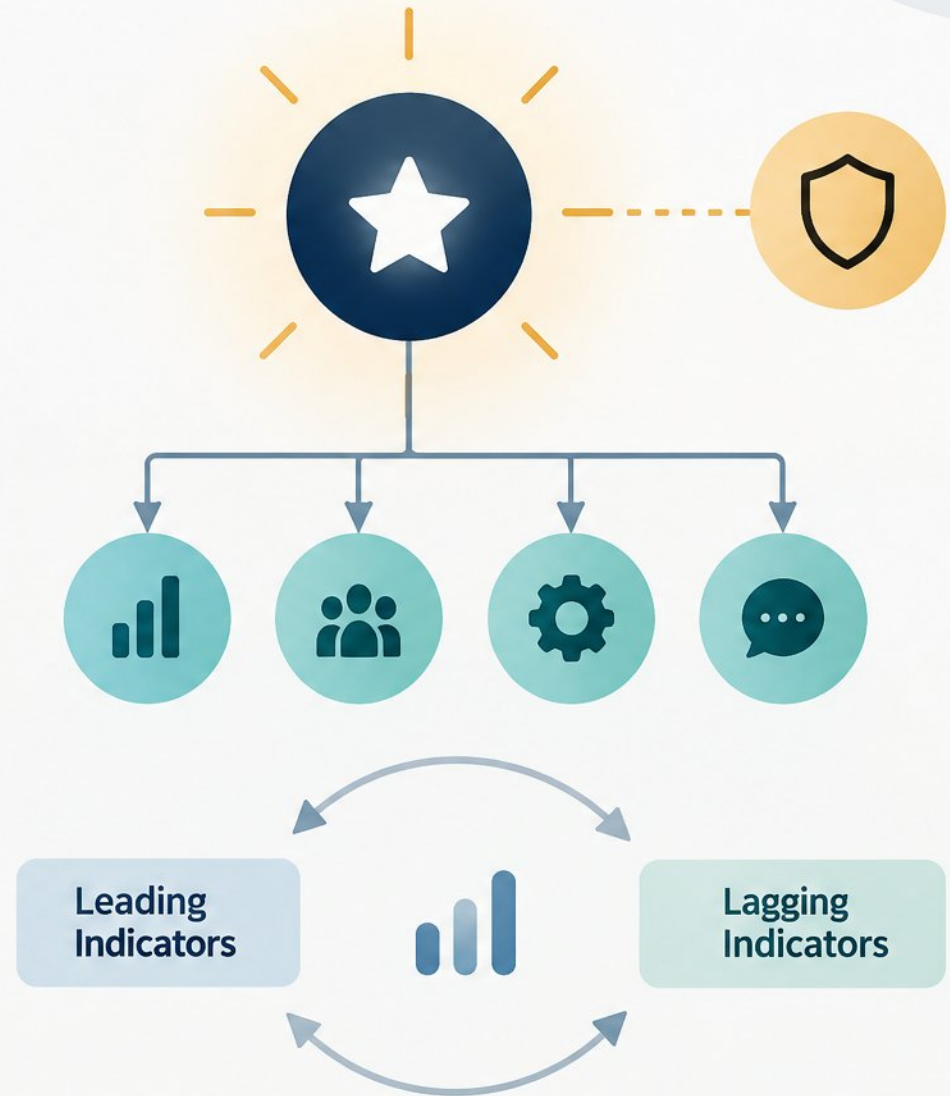
Sprint Backlog



SMALL EXPERIMENTS
BIGGER PROGRESS

Metrics and Evidence of Product Outcomes

- One North Star Metric expresses core customer value, leading revenue
- Good North Stars reflect value, lead revenue, stay actionable, resist gaming
- Break the North Star into 3–5 input metrics teams own weekly
- Pair the North Star with a counter-metric guardrail against Goodhart's Law
- Track leading and lagging indicators: cycle time plus adoption and retention
- Review quarterly; revise only when product, market, or model materially changes



Operating Rhythms and Stakeholder Communication

ALIGN
LEARN
DECIDE
TOGETHER



Every touchpoint should help the organization choose, learn, or remove a constraint



Weekly outcome review: did our bets change behavior, and what next?



Sprint review is a learning checkpoint, not a demo ceremony



Decision log and lightweight governance keep decisions visible



Match cadence to how fast each stakeholder's confidence decays



One source of truth, tailored views, not separate status meetings



Scaling Product Practice Across Teams



Complexity compounds:
20 teams create 190 possible
inter-team dependencies



Tiered decision rights:
teams own local calls,
executives own irreversible bets



Cut coordination cost by
organizing around outcomes,
not more process



Platform teams need dedicated
product ownership, or downstream
teams starve



Fastest teams at scale do fewer
things; focus beats headcount



AI, Judgment, and the 2026 Product Context



AI compresses discovery and delivery; the bottleneck moves from building to deciding



PM's job: ensure the right person decides, on evidence, before the window closes



Directed autonomy: define constraints, decision rights, and governance before AI-accelerated work



Product engineer model: technical implementation, product judgment, user empathy, business fluency



Governance calibrated to discovery velocity; pre-sprint constraint reviews and artifact-level ownership



Use AI for synthesis and prototypes; keep human judgment on outcomes and trade-offs



Common Pitfalls and How to Course-Correct

Common Pitfalls



Agile as ceremonies, not a learning system



Discovery skipped; only output measured, so nobody uses features



Strategy split from execution, creating proxy owners



Vanity metrics rise without predicting retention or revenue



Process scaled before fixing strategy, ownership, customer proximity

How to Course-Correct



Fix: tighten feedback loops, clarify decision owners, cut untraceable work

Your 30-60-90 Day Action Plan

From insight to impact — build the habits, make progress, and set up what's next.

Days 1–30



Days 1–30: map feedback sources; pick one product area and outcome



Days 1–30: one customer conversation weekly; start opportunity solution tree



Days 31–60



Days 31–60: reframe a roadmap item as an outcome; run one small experiment



Days 31–60: launch weekly outcome review and decision log with owners



Days 61–90



Days 61–90: review what changed, prune what failed, name next constraint



Build leadership support with evidence of learning, not delivery status

DISCOVER
FOCUS
MAKE PROGRESS
WHAT'S NEXT

PersonWise.

PersonWise • AI digital-human course platform



Scan the code or click anywhere to watch this course live, with voice Q&A

personwise.ai/t/7180d65f/public