

The Outcome-Based Agile Framework

In a world where agile has often been diluted into process rituals without purpose, the *Outcome-Based Agile Framework* reclaims its core intent: to create meaningful change through adaptive learning and value delivery. Inspired by the original [Agile Manifesto](#), this framework defines a model where teams are driven by **outcomes**, **not requirements**, and guided by evidence, not assumption.

> **Common Misreadings**

- >
- > - **"No Upfront Requirements"** does **not** mean chaos. It means starting with **intent**, not **specification**.
- > - **"Governance exists to support autonomy"** does **not** imply lack of structure. It means governance should **enable**, not **control**.

Manifesto

We are uncovering better ways of creating value by focusing on outcomes rather than outputs. Through this work, we have come to value:

- **Outcomes over requirements**
Because solving problems matters more than delivering predetermined solutions.
- **Constraints over scope**
Because framing the problem is more productive than prescribing its answer.
- **Discovery over certainty**
Because we can't predict value—we must find it through learning.
- **Accountability to results over compliance to plans**
Because teams should be responsible for making change, not just delivering work.
- **Alignment on intent over agreement on features**
Because shared goals outlast specific ideas.

That is, while there is value in outputs, defined scopes, and planned features, **we value the ability to learn and adapt toward real outcomes far more.**

Framework Tenets

1. No Upfront Requirements

Work does not begin with a list of features, specifications, or outputs. Teams begin with problems to solve and outcomes to achieve. The only fixed inputs are constraints.

2. Clarify the Purpose of the Outcome

Ensure the team understands why the outcome matters—its strategic, operational, or user-centered importance—before starting the work. Quality attributes often define what "good" looks like for a given outcome. Teams should clarify which attributes (e.g., usability, reliability, performance, security) are essential for success, and ensure those are explicitly part of the intended impact.

3. Problem Ownership Over Task Execution

Teams are given ownership of a problem and trusted to determine how best to solve it within the defined boundaries.

4. Outcomes Are Observable Changes

An outcome is a measurable shift in user behavior, business value, or system performance. Delivery is only valuable if it contributes to these shifts. This includes improvements in non-functional areas like usability, system reliability, deployment efficiency, and security. Quality attributes must be observable and verifiable as part of outcome validation.

5. Constraints Define Limits, Not Solutions

Constraints—technical, legal, ethical, or otherwise—are real and respected. They inform exploration but must not dictate the final form of a solution.

6. Continuous Discovery Is Mandatory

Discovery and delivery happen together. Teams explore problems, test assumptions (solution hypotheses), and validate ideas in real time. This includes discovering which quality attributes matter most in the context and validating them with users or stakeholders continuously.

7. Evidence Is the Arbiter

All decisions are made based on learning from real users and data. **Plans are hypotheses**, not contracts.

8. Strategy Is Intent, Not Instruction

Leaders set direction through vision and desired outcomes, not through detailed feature roadmaps.

9. Simplicity of Governance

Governance exists to support autonomy and learning. It must be light, minimal, and enabling—never controlling.

10. Regular Blameless Retrospectives Through After Action Reviews (AAR)

Teams must conduct regular retrospectives at intervals appropriate to their context (after iterations, deliveries, daily, weekly, or combinations). The recommended format is the *After Action Review* (AAR), which uses four blameless and structured questions:

- **What was supposed to happen?** (What was planned?)
- **What happened?** (What actually occurred?)
- **Why did it happen?** (Root causes and contributing factors)
- **What did we learn?** (Key insights and adaptations)

Each question is discussed separately and sequentially by the entire team. Conversations about "what was planned," "what happened," and "why" must be isolated from each other to avoid conflating facts and analysis. Learning points can be documented at any time in the "what did we learn" section.

Differences in team members' understanding are celebrated as learning opportunities rather than treated as errors. The goal is not for all team members to have identical views but to explore the richness of different perspectives.

It is considered rude to interrupt team members while they are speaking. The facilitator should ensure that each team member has the opportunity to speak without interruption and can finish sharing their thoughts before others respond.

Importantly, the "what did we learn" section, including any optional recording in a knowledge base, must never include the names of individual team members. Only lessons learned, without attribution to specific people, are captured to ensure a fully blameless environment that fosters openness and continuous improvement.

How to Apply This Framework

> **Start small:** Before applying the framework to any initiative—whether one, many, or all—begin with the Readiness Assessment in the next section. It will help you evaluate whether your teams, culture, and leadership are prepared for outcome-driven ways of working. The results will highlight strengths to build on, surface gaps to address, and guide a pace of adoption that fits your context—ensuring the framework supports rather than overwhelms.

Organizations adopting Outcome-Based Agile should begin by clearly defining the outcomes they seek. These outcomes should not only be real, measurable changes in user or business behavior but also have a clearly articulated purpose. Clarifying *why* each outcome matters helps ensure alignment across teams, guides prioritization, and connects day-to-day work to broader organizational goals.

In defining and pursuing outcomes, teams should explicitly consider which quality attributes are critical to achieving the desired change. These attributes—such as security, suitability, usability, reliability, maintainability, compatibility, or deployment efficiency—must be treated as part of the outcome, not as secondary requirements. Quality attributes should be framed as hypotheses to test, measured as part of delivery, and evaluated through direct feedback and evidence, not assumed as implicit side effects of feature development.

Teams should be cross-functional, capable of discovery and delivery, and given end-to-end responsibility for their outcomes.

Roadmaps become hypothesis backlogs. Requirements become constraints. Progress is measured not by story points, but by progress toward the desired change.

Leadership must shift from steering work to enabling learning. Teams must shift from implementing solutions to exploring possibilities. And the organization must create space for evidence to matter more than certainty.

Recommended Implementation

This framework **recommends—not prescribes—a structured yet adaptable approach** to outcome-driven planning and experimentation. It provides a practical system for making not just work visible, but also for surfacing **intent, constraints, signals, learnings, and measurable results.**

At its core is a **1+5 nested loop**, designed to align rapid action with reflective learning. This structure guides teams through continuous cycles of intent-setting, hypothesis-building, execution, and adaptation—without imposing rigid process overhead.

The six parts of the loop are:

0. Establish Feedback Loops Early
1. Create an Outcome Card
2. Initialize an Experiment Canvas
3. Plan Using the Recognitional Planning Model (RPM)
4. Translate CoA into Executable Tasks via an Exploratory Kanban Board
5. Facilitate After Action Reviews (AARs)

0. Establish Feedback Loops Early

Before initiating any outcome or experiment, prioritize the creation of **instant or near-instant feedback mechanisms** that are specific to your domain and operational context. These loops form the backbone of your ability to validate decisions in real-time and to pivot based on emerging insights.

Without timely feedback, experimentation risks becoming guesswork. Feedback loops ensure that learning is continuous, grounded, and actionable.

> *Tip: Consider making this one of your first Outcome Cards—ensuring early visibility into what's working and what's not.*

1. Create an Outcome Card

Start by defining a clear, focused **intent** that communicates the outcome you aim to achieve. The **Outcome Card** acts as a central alignment artifact—anchoring planning, experimentation, and execution in a shared understanding of purpose, constraints, and context.

The Outcome Card should be concise yet rich enough to guide action and adaptation. Populate it with the following elements:

- **Context** - The situational background or environmental conditions that give rise to the need for action. This ensures that all stakeholders understand the circumstances under which the intent is being pursued.
- **Intent** - Expressed as a combination of:
 - **End-State** - The desired result or condition to be achieved.
 - **Constraints** - Boundaries that must not be violated (e.g., safety, policy, cost).
 - **Purpose (IOT)** - The strategic or operational rationale behind the intent; typically phrased as "in order to..."
 - > **Example:**
 - > *Intent: "Prevent the fire from reaching the gas station (end-state), without risking firefighter safety (constraint), in order to maintain critical infrastructure and avoid civilian casualties (purpose)."*
 - >
 - > *Tip: Use the full intent statement as the title of the Outcome Card. This helps keep constraints visible and prevents them from silently transforming into assumptions or rigid requirements.*
- **Signals of Success** - Preliminary indicators—quantitative or qualitative—that suggest progress toward the end-state. These can evolve over time but serve as an early compass for situational awareness and adaptation.
- **Quality Attributes** - Non-functional or qualitative properties that help define what "good" looks like for the outcome. Examples may include **resilience**, **scalability**, **safety**, or **ease of use**, depending on the domain.
- **Owner** - The team accountable for exploring and delivering this outcome. This provides a clear point of ownership for planning, execution, and follow-through.
- **Target Timeframe** - A flexible but bounded estimate for when meaningful signals or results should emerge. It helps scope the experiment and prevent stagnation, while leaving room for complexity and adaptation.
- **Supporting Outcomes** - Outcomes that contribute to the achievement of this primary intent. If an Outcome Card includes supporting outcomes, it is considered a **north-star Outcome Card**—typically used at the departmental, portfolio, or organizational level to anchor strategic direction.

A **north-star Outcome Card** should be **broad in definition** and **non-prescriptive**—avoiding early commitments to specific technologies, solutions, or metrics. Its role is to align diverse efforts without constraining how they're executed.

North-star Outcome Cards **do not require an Experiment Canvas**, though they may include one if exploration or iteration at the strategic level is needed. Their progress is often tracked through high-level **signals of success** and realized through the coordinated impact of supporting outcomes.

Each **supporting outcome** must be documented in its own Outcome Card and **must include an associated Experiment Canvas**. This ensures that supporting outcomes are actionable, testable, and owned—allowing teams to iterate, adapt, and deliver in alignment with the broader intent.

> *Tip: Use supporting outcomes to translate strategic goals into concrete work across teams—maintaining clarity of direction while enabling decentralized execution.*

2. Initialize an Experiment Canvas

After defining your Outcome Card, the next step is to set up an **Experiment Canvas**. This canvas serves as the operational bridge between **intent** and **execution**, capturing the hypothesis, tasks, and learning that emerge throughout the experimentation cycle. It is designed to evolve over time and supports structured reflection, making learning explicit and reusable.

The OBAF-format Experiment Canvas is made up of seven hexagons—six arranged in a circle around a central one. Each outer hexagon connects directly to the center, which anchors the experiment to its strategic outcome. The outer hexagons are positioned like points on a clock face, and moving clockwise through them mirrors the natural flow of running an experiment—from forming a hypothesis to capturing results and learnings. This structure helps keep every part of the experiment aligned with its intended outcome and purpose.

1. Center Hexagon - Outcome Definition

At the center of the canvas is the outcome definition, representing the strategic objective the experiment is ultimately serving. This hexagon contains the linked **Outcome Card**, which anchors the work to the broader outcome portfolio and provides traceability between the strategic goals and the experiment being conducted. Everything else in the canvas connects back to this core definition, maintaining alignment with the intended impact.

2. 12 o'clock - Hypothesis (Course of Action)

At the top of the canvas is the hexagon that holds the hypothesis under test. This is expressed as a *Course of Action* (CoA), developed during the RPM planning session. The hypothesis should be a clearly stated, testable assumption—describing what you expect will happen if a particular action is taken. It links a proposed intervention to an anticipated outcome, allowing the team to validate or challenge that assumption through experimentation. To provide clarity and structure, the Course of Action is ideally broken into three phases: **initially**, to outline the first steps; **thereafter**, to describe the follow-on actions; and **finally**, to define how the effort will conclude or transition.

> *Leave this blank until the planning session has produced a CoA.*

3. 1:30 - Evaluation Criteria

Moving clockwise, the next hexagon defines the evaluation criteria for the experiment. This includes the thresholds, conditions, or boundaries that determine whether the hypothesis can be considered valid. It sets the standards against which success will be judged, such as acceptable tolerances or target ranges, and is critical for determining when to pivot, persevere, or stop the experiment altogether.

4. 4:30 - Metrics

The next hexagon contains the metrics that will be observed and measured throughout the experiment. These signals provide evidence of whether the intended outcomes are being achieved. Metrics may be quantitative, such as performance indicators or usage statistics, or qualitative, such as user behavior or feedback themes. They should be directly relevant to both the evaluation criteria and the defined outcome, providing insight into whether progress is being made.

> Optionally, leave this blank until after the planning session.

5. 6 o'clock - Experiment Steps

At the bottom of the canvas are the experiment steps, which describe the concrete actions that will be taken to test the hypothesis. These steps are based on the Course of Action and typically translate into tasks managed through the **Exploratory Kanban** board. The steps should be practical and testable, enabling focused execution while allowing for rapid learning through iteration.

> Leave this blank until after the planning session.

6. 7:30 - Outputs

Continuing clockwise, this hexagon lists the outputs generated during the experiment. These are tangible artifacts or deliverables that result from executing the steps of the Course of Action. Outputs might include code commits, releases, deployments, documentation, prototypes, decisions, or other forms of evidence that something has been built, tested, or delivered as part of the experiment.

7. 10:30 - Results and Learnings

The final hexagon captures the results and learnings from the experiment. This space is progressively populated through **After Action Reviews (AARs)**, which reflect on what occurred during execution. It documents the observed outcomes, whether expected or not, and records the insights gained. Learnings may come from both success and failure and are essential for refining future hypotheses or adjusting strategy.

3. Plan Using the Recognitional Planning Model (RPM)

Conduct a focused planning session using the **Recognitional Planning Model**, which supports rapid, experience-driven decision-making—especially under uncertainty, time pressure, or incomplete information. Rather than evaluating multiple alternatives, RPM relies on pattern recognition to generate a single, viable **Course of Action (CoA)** based on the decision-maker's mental model of the situation.

This CoA becomes your **operational hypothesis**, a testable narrative of how the outcome will be achieved under the given constraints and context.

Update the **Experiment Canvas** with the following elements:

- **Hypothesis CoA** - Capture the planned course of action in a narrative format, ideally structured as a three-phase progression: *"Initially, we...; thereafter, we...; and finally, we..."*
This storytelling form improves clarity, alignment, and memory recall—especially in complex or time-critical operations.
- **Evaluation Criteria** - Define what success looks like by specifying thresholds, tolerances, or boundaries that must be met for the outcome to be considered achieved.
- **Metrics** - Identify the signals—both quantitative and qualitative—that will be used to track progress and validate whether the intended effects are occurring.
- **Falsifiability Check** - Confirm that the hypothesis is **testable and falsifiable**, ensuring that failure to achieve the expected outcome can be clearly recognized and learned from.

4. Translate CoA into Executable Tasks

Convert the CoA into a set of small, testable **execution tasks**. These tasks enter the **Exploratory Kanban Board**, where the focus is on maintaining momentum while preserving the ability to learn and adjust. The board is optimized for flow and responsiveness, supporting:

- **Limited batch size** - Prevent buildup of large queues and reduce overloaded work-in-progress (WIP).
- **Small work items** - Scope tasks tightly to enable rapid feedback and easier course correction.
- **Continuous integration** - Ensure that insights, deliverables, and results are continuously folded back into both the **Experiment Canvas** and any relevant technical workflows, such as code merges and deployments.

5. Facilitate After Action Reviews (AARs)

Schedule **regular After Action Reviews**, or trigger them contextually—for example, when the backlog runs thin or a natural breakpoint occurs in execution. AARs are essential for institutionalizing learning and deciding whether to reinforce, adjust, or abandon a given CoA.

In each AAR:

- Compare outcomes to the hypothesis
- Assess what succeeded, failed, or surprised
- Update the **Experiment Canvas** with:
 - Refined metrics and indicators
 - New contextual insights
 - Results and documented learning

These updates may inform refinements to the current Outcome Card or trigger a new iteration of the planning and experimentation cycle.

Recognitional Planning Model

The **Recognitional Planning Model (RPM)** is a cognitive decision-making framework originally developed within military and emergency response contexts, particularly in high-stakes environments where rapid yet effective decisions are needed despite uncertainty or incomplete information. Its foundations trace back to the work of Gary Klein and others studying **naturalistic decision-making**—how experts make fast, effective choices based on experience rather than exhaustive analysis.

The term "recognitional" reflects the core mechanism of the model: rather than generating and comparing multiple detailed options from scratch (as in traditional decision theory), experienced practitioners **recognize familiar patterns or situations**, which trigger **pre-formed mental models or scripts** for what to do. These scripts are then quickly evaluated for fit. If they seem workable, they are executed with minimal delay. If not, they are adapted or rejected, and another option is tried. In short, it's not about calculating the best plan, but identifying a "good enough" one based on what's known right now.

OBAF (Outcome-Based Adaptive Framework) draws from RPM to empower teams to **move from strategy to action** efficiently without being paralyzed by the need for perfect plans. In a self-organizing, cross-functional team, RPM helps teams generate a **Course of Action (CoA)** rooted in shared understanding, lived experience, and the situational context—rather than top-down directives or speculative analysis.

RPM is especially well-suited for environments where:

- Conditions are changing or ambiguous.
- Information is incomplete but action is still required.
- The team has enough collective experience to recognize useful patterns and plausible next steps.
- Rapid experimentation and adaptation are valued over rigid execution.

When applied to planning within an OBAF context, RPM supports the creation of an experiment-ready CoA that is simple, realistic, and immediately actionable. It enables teams to move quickly toward testing while maintaining alignment with strategic outcomes.

How to Apply RPM in a Planning Session

When a team applies RPM, they are not brainstorming a long list of potential ideas to analyze and debate. Instead, they begin by identifying **what is known**, **what's being observed**, and **what similar situations the team has seen before**. From this, a plausible Course of Action naturally emerges—a storyline of what to try next, based on judgment, relevance, and feasibility.

The result is a CoA written in a storytelling format, broken into three distinct phases:

Initially

This phase sets the immediate next steps—the first actions the team will take based on what is currently known and achievable. It may involve establishing a starting point, setting up conditions for experimentation, or initiating a change. These steps should be clear, specific, and focused on creating early signals or momentum.

> Example: “Initially, we will release the new onboarding prompt to 10% of users on the signup page to observe whether it improves progression to the dashboard.”

Thereafter

This is the unfolding middle of the story—the reaction phase. It outlines how the team will follow up based on initial feedback or results, what further actions will be taken, and how the hypothesis will evolve through implementation.

> Example: “Thereafter, we will compare engagement metrics between the test and control groups and run interviews with selected users who completed the new flow to understand their experience.”

Finally

This phase closes the loop. It describes the conditions or criteria for wrapping up the experiment, scaling the intervention, or shifting focus. It might include what will happen if the hypothesis is confirmed or disproven, or how the team will capture and share learnings.

> Example: “Finally, if we see a 20% or higher increase in dashboard activation within the test group, we will roll out the prompt to all users and update our user journey map to reflect this new entry pattern.”

The Cognitive Engine

RPM provides a pragmatic, grounded approach to planning—especially suited to autonomous, cross-functional teams working under uncertainty. By focusing on recognition, experience, and iterative adaptation, it enables teams to act with clarity and purpose. In OBAF, it is the cognitive engine behind the Course of Action, helping turn strategic intent into experiment-ready hypotheses that are both testable and actionable.

Exploratory Kanban

To support daily exploratory work in Outcome-Based Agile, teams are encouraged to adopt a lightweight Kanban system that promotes focus, flow, and learning across disciplines. This system is not limited to software—it applies to any kind of outcome-focused work, including research, operations, design, service development, or policy.

Suggested Columns

- **Ready for Development** - Items selected from the backlog, framed as hypotheses to explore
- **In Development** (or **Doing**, **In Process**, etc) - Items currently in active exploration, design, testing, or creation
- **Ready to Test** - Work paused for review or integration, awaiting evaluation
- **Test** - Evaluation of fitness, coherence, quality attributes, or integration with other efforts
- **Done** - Complete and ready for delivery, use, or deployment

A **Backlog** (or **To Do**) column contains the team's working hypotheses. These are not verified answers but informed bets. When the backlog becomes too shallow, it triggers a discovery or planning session to replenish ideas grounded in the outcome's intent and constraints.

Pull System Across the Flow

The system operates as a pull-based workflow:

- **Work is never pushed forward.** Instead, each column **pulls** from the previous one when capacity allows and context is ready.
- When work in the **In Development** (or **Doing**) column is complete, it is moved to **Ready to Test**, not directly into **Test**. This ensures it is **explicitly pulled into validation** when the team has capacity and focus.
- Similarly, teams **pull work from the Backlog into Ready for Development**, ensuring prioritization is intentional and capacity-aware.

This pull mechanism prevents overload, respects WIP limits, and encourages reflection at each transition. It reinforces that movement between phases is a deliberate choice, not an automatic step.

WIP Limits

To support team autonomy, sustainable pace, and continuous learning, the columns **Ready for Development**, **In Development**, and **Test** should include **Work In Progress (WIP) limits**. These are flexible boundaries set by the team to reveal bottlenecks, maintain smooth flow, and foster intentional decision-making—rather than rigid rules imposed externally.

Validation Happens Outside the Board

This board reflects internal work readiness and coordination—not outcome success. Actual validation of a hypothesis happens through **external metrics**, preferably **automated and continuous**, such as behavior change, system performance, or user engagement. The **Done** column indicates readiness for delivery, not proof of effectiveness.

Why This Flow Supports OBAF

This Exploratory Kanban system enables:

- **Visible exploration** of hypotheses
- **Team-managed flow** based on capacity, not forced schedules
- **Clear separation of working and validating**
- **Adaptive prioritization** grounded in outcomes
- **Continuous discovery** through parallel metrics and learning

It supports autonomy, encourages sustainable pace, and keeps delivery aligned with real-world evidence—as Outcome-Based Agile demands.

Validation Outside the Board

The Kanban board reflects the team's internal readiness and coordination—not the validation of actual outcomes.

- **Validation happens beyond the board**, through continuous, preferably automated, metrics (e.g., behavior change, system performance, user engagement).
- There are no predefined validation criteria before delivery, because outcomes cannot be fully anticipated.
- **Validation is emergent**, based on real-world feedback and production use—via A/B tests, before-after comparisons, or system telemetry.

Equally important, OBAF emphasizes qualitative validation. Insights from user interviews, observational research, session recordings, and open-ended feedback provide context and meaning that quantitative signals alone cannot capture. These methods reveal why users behave the way they do, not just what they do. Emerging technologies—such as AI-powered sentiment analysis, natural language processing, and real-time experience tracking—now make qualitative validation more scalable and actionable than ever before.

The Outcome-Based Agile Framework treats both forms of evidence as essential and complementary. Effective validation blends data and dialogue, measuring behavior while understanding intent. This ensures that outcomes are not only observable but meaningful.

Common Anti-Patterns in Outcome Validation

To maintain the integrity of outcome-based work, teams must be mindful of common traps that quietly erode learning and agility. These anti-patterns often emerge when metrics are misused, constraints become overly rigid, or activity is mistaken for value.

1. Goodhart's Law in Action

> When a measure becomes a target, it ceases to be a good measure.

When teams fixate on hitting specific numbers (e.g., reducing bounce rate or increasing click-through), they may lose sight of the underlying change that truly matters. Metrics should serve as signals for exploration—not targets to hit at any cost.

2. Vanity Metrics

Not all data is meaningful. Metrics like page views, impressions, or internal velocity can create the illusion of progress while masking stagnation. These vanity metrics look good on dashboards but rarely reflect real user or business outcomes. Prioritize signals that tie directly to meaningful behavior change or value realization.

3. Constraint Creep

Constraints exist to define safe boundaries—not to prescribe solutions. Over time, teams often inherit outdated specifications or interpret vague standards as fixed requirements. This slow expansion of what's considered "non-negotiable" can quietly kill innovation. Revisit constraints regularly to ensure they're still valid, contextual, and evidence-based.

What to do instead:

- Use metrics as **learning tools**, not success criteria. Let signals inform, not dictate.
- Question the purpose behind each measurement. Ask, *"What does this really tell us?"*
- Revalidate constraints. Treat them as hypotheses too—especially when they block experimentation.

These reminders help teams stay focused on learning and progress that matters—so validation remains a discovery process, not a performance ritual.

Outcome vs. Output

A frequent challenge in applying outcome-based thinking is the **blurring between outcomes and outputs**, especially at the team level. Outputs are **things we deliver**—features, services, improvements. Outcomes are the **observable changes** those outputs create in the real world.

- > An output is **delivered**.
- > An outcome is **validated** by real-world evidence.

Teams often mistake improved functionality (e.g., "faster page load") for outcomes, when the true goal is behavioral or value-based (e.g., "more users complete the checkout process").

Examples

- Outcome:** Users complete tasks faster
Signals of change: Higher task completion rates and a reduction in user drop-offs
Related outputs: Page speed improvements and a streamlined user interface with fewer steps
 - Outcome:** More users set up their accounts through self-service
Signals of change: Increased percentage of accounts created without needing human assistance
Related outputs: Redesigned onboarding experience and automated support tools
 - Outcome:** Fewer users require support for password issues
Signals of change: Lower volume of support tickets and positive user feedback
Related outputs: Enhanced "Forgot Password" functionality and better input validation
 - Outcome:** Users trust billing processes more
Signals of change: Fewer billing-related complaints and higher Net Promoter Score (NPS)
Related outputs: Clearer invoice layouts and helpful usage explanations via tooltips
 - Outcome:** Increased usage of the scheduling feature
Signals of change: Higher adoption and more frequent usage of the feature
Related outputs: Launch of the feature, onboarding guidance, and supportive help content
- > **Example:**
 - > "Improve page load time" is an **output** (a quality attribute). "Increase checkout completion rate" is the **outcome**. The former contributes to the latter, but **is not the outcome itself**.

Tip for Teams

When defining an outcome, ask:

- What will people **do differently** if this works?
- How will we **observe or measure** that change?
- Could the outcome be achieved **in multiple ways**?

Outcomes describe **why something matters** and how success is observed—not what to build.

Cross-Team Coordination and Outcome Ownership

Outcomes should not be split across teams with different priorities. To ensure coherence and ownership:

- Each outcome should belong to one clearly defined team or pod.
- If multiple teams contribute, they must act as a single outcome-focused unit.
- Organizational structures should evolve to reflect outcome boundaries (Conway's Law in reverse).
- Interfaces and dependencies should be framed as contracts, not coordination burdens.

Teams are encouraged to reorganize when outcome ownership becomes diluted or coordination overhead increases.

Leadership and Oversight

Leadership in OBAF means enabling, not directing. Oversight should be nearly invisible:

- **Optimal oversight is automated:** KPIs, usage data, or real-world impact signals.
- Leaders guide through vision and constraints—not feature lists or status reports.
- Interventions should only occur in cases of systemic failure, ethical risk, or learning breakdowns.
- Sponsors and managers should practice servant leadership – understood here as a facilitative approach – by funding experiments, supporting outcome framing, and modeling evidence-based decision-making.

From Status-Reporting to Evidence-Framing

For leaders and sponsors, shifting from traditional output oversight to outcome enablement means more than changing what gets tracked—it requires changing how conversations happen.

In status-reporting cultures, reviews often focus on surface-level indicators: percent complete, story points burned, or tasks delivered. These measures are easy to collect but rarely illuminate whether meaningful progress is happening.

Evidence-framing transforms these conversations. Instead of asking, *"Are we on track?"* leaders ask, *"What have we learned?"*, *"What signals are emerging?"*, and *"What's the current level of confidence in the outcome?"*

This shift happens progressively, often in small steps:

- A delivery update evolves into a learning review, where the team shares new insights about user behavior or system feedback—not just what was built.
- Instead of tracking feature completion, leaders start watching for real-world signals that value is emerging (e.g., user adoption, friction reduction, behavior change).
- Weekly reports stop listing tasks and start summarizing discoveries, test results, and adjustments based on evidence.
- "Red-yellow-green" status summaries give way to qualitative confidence levels—rooted in both data and team insight.

As leaders adopt this posture, they stop asking for certainty and start investing in clarity. They stop steering by roadmap and start enabling exploration, grounded in trust and observable impact.

The goal is not to eliminate accountability—but to make it meaningful. When teams are asked to show **evidence of learning**, not just activity, accountability becomes a tool for alignment, not control.

Outcome-Based Agile Readiness Assessment

This *Readiness Assessment* focuses on organizational, team, and leadership readiness across five key dimensions. Each area includes a short description and 3 yes/no questions. Use it as a kickoff diagnostic or self-check before adopting the framework.

1. Outcome Thinking

Are we focused on value and behavior change, not just delivery?

- [] Do we define success in terms of user behavior, business value, or system performance—not just features delivered?
- [] Do we regularly ask “*Why does this matter?*” before “*What are we building?*”
- [] Do teams have a clear understanding of the desired outcome before starting work?

> If “yes” to 2 or more: Outcome awareness is emerging.

2. Team Autonomy and Cross-Functionality

Can teams own the problem, not just execute tasks?

- [] Do teams have the skills to explore and deliver without constant handoffs?
- [] Are teams trusted to make solution decisions within clear boundaries?
- [] Do teams work from outcomes or constraints, rather than prewritten tickets?

> If “yes” to 2 or more: Autonomy foundations are in place.

3. Evidence-Driven Culture

Are decisions grounded in learning from users and data?

- [] Do we treat plans as hypotheses that can change based on new learning?
- [] Are experiments, metrics, or real-world signals used to guide work?
- [] Do leaders welcome evidence that challenges assumptions?

> If “yes” to 2 or more: Culture supports evidence-based work.

4. Psychological Safety and Learning

Can people speak up, learn from failure, and improve continuously?

- [] Are retrospectives or reviews blameless, structured, and regularly held?
- [] Can team members safely admit mistakes or raise concerns?
- [] Are failures framed as learning opportunities, not performance gaps?

> If “yes” to 2 or more: A learning environment exists.

5. Leadership as Enabler

Do leaders guide through intent and remove blockers?

- [] Are leaders setting direction through desired outcomes—not features or task lists?
- [] Do managers act more as sponsors and coaches than controllers?
- [] Is governance light, adaptive, and supportive of learning and change?

> If “yes” to 2 or more: Leadership is aligned with OBAF principles.

Scoring Summary

- **4-5 areas with "2 or more YES answers"**: You're ready to pilot The Outcome-Based Agile Framework.
- **2-3 areas**: Start with a small team or experiment and invest in coaching.
- **0-1 areas**: Begin with mindset and cultural groundwork before rollout.

Facilitation Guide

1. **Kickoff Workshop**: Define the outcomes, clarify the purpose behind each outcome, surface constraints, and establish team autonomy
2. **Cadence Design**: Establish short cycles of delivery and feedback (e.g., 1-2 weeks)
3. **Feedback Loops**: Embed continuous discovery methods (user interviews, A/B tests, analytics reviews)
4. **Review Rituals**: Use retrospectives in the form of AARs to focus on what was planned, what happened, why it happened, and what was learned (see Tenet 10 for full AAR structure)
5. **Leadership Coaching**: Train sponsors and managers to support outcome framing, not output control

Simple Process Overview

1. **Input**: Outcome + Constraints
2. **Process**: Discovery + Delivery (Iterative)
3. **Output**: Validated Experiments, Prototypes, and Releases
4. **Outcome**: Real-world change (measured)
5. **Feedback Loop**: After Action Reviews (AARs) inform learning and next steps

Key Reminders

- If you are writing a solution before exploring the problem, pause.
- If requirements are treated as fixed outputs, reframe them as boundaries.
- If retrospectives assign blame, reset the culture.
- If decisions ignore evidence, restart the discovery.

Checklists

These ten checklists—each with five concise, actionable points—are designed to support kickoff meetings, initial workshops, or the launch of new outcomes within established teams. They help ensure clarity, alignment, and momentum across a range of critical topics:

1. Clarity and Alignment on Outcomes
2. Lightweight Outcome Metrics
3. Discovery Embedded in the Work
4. Teams Own Outcomes, Not Tasks
5. Reframing Requirements as Constraints
6. Structuring After Action Reviews (AARs)
7. Governance That Enables, Not Controls
8. Organizing Around Shared Outcomes
9. Leadership as Outcome Enablers
10. Organizational Agility Support

1. Outcome Definition and Framing

- [] Is the outcome expressed as a change in behavior, business result, or system capability?
- [] Has the *purpose* of the outcome been made explicit (e.g., why it matters to users, teams, or strategy)?
- [] Have the relevant quality attributes been identified (e.g., usability, security)?
- [] Is the outcome flexible in *how* it's achieved, but firm in *why* it matters?
- [] Have success signals (qualitative or quantitative) been defined?

2. Measurement and Evidence

- [] Is there a measurable signal for outcome progress (e.g., a proxy, heuristic, or direct metric)?
- [] Can the signal be tracked continuously or in short cycles?
- [] Are stakeholders aligned on what "evidence" looks like?
- [] Are teams using the data to adjust their approach, not just report status?
- [] Are metrics viewed as signals to explore, not KPIs to hit blindly?

3. Continuous Discovery Integration

- [] Are discovery activities happening in parallel with delivery?
- [] Is learning from experiments (e.g., A/B tests, prototypes) captured?
- [] Do discovery insights influence next steps or pivot decisions?
- [] Are assumptions treated as hypotheses, not facts?
- [] Are users and stakeholders engaged early and often?

4. Team Autonomy and Problem Ownership

- [] Do teams start with a problem, not a backlog of predefined tasks?
- [] Can teams adjust scope and solutions to achieve outcomes?
- [] Are cross-functional skills available within the team?
- [] Is problem framing part of team planning?
- [] Are teams trusted to challenge the framing of outcomes if needed?

5. Constraints and Boundaries

- [] Have non-negotiable constraints (technical, regulatory, ethical) been documented?
- [] Are constraints used to shape exploration, not dictate outputs?
- [] Is there a shared understanding of what can and cannot change?
- [] Are teams enabled to discover the best solution within those constraints?
- [] Do constraints evolve with discovery if context changes?

6. Blameless Retrospectives and Learning Culture

- [] Are retrospectives structured around *What was planned? What happened? Why? What did we learn?*
- [] Is participation inclusive and non-hierarchical?
- [] Are learnings documented without names or blame?
- [] Are multiple perspectives explored without forcing consensus?
- [] Do learnings influence future behavior or decisions?

7. Governance and Simplicity

- [] Are governance practices light and fit-for-purpose?
- [] Are teams free to make technical and design decisions within boundaries?
- [] Is governance focused on learning, not compliance?
- [] Are outcome reviews prioritized over milestone checklists?
- [] Are governance structures adaptive as context changes?

8. Coordination Across Teams

- [] Are outcomes divided in a way that minimizes cross-team dependencies?
- [] Do teams exposing interfaces (e.g., APIs, services) treat them as contracts?
- [] Is communication structured to match system architecture (Conway's Law)?
- [] Are shared discovery efforts conducted for cross-cutting concerns?
- [] Are teams encouraged to re-org when friction emerges?

9. Leadership and Sponsorship

- [] Are leaders setting vision and outcomes, not dictating features?
- [] Do they reward learning, even when it disproves assumptions?
- [] Are they present in retrospectives or discovery reviews?
- [] Are managers being coached on adaptive leadership practices?
- [] Do they protect space for teams to explore and learn?

10. Cultural Readiness and Adaptation

- [] Is psychological safety actively cultivated?
- [] Are success and failure both treated as learning opportunities?
- [] Are language and metaphors aligned to outcome thinking (e.g., "What problem are we solving")?
- [] Is process adaptation encouraged over adherence to rituals?
- [] Is evidence given more weight than authority or tradition?

Applying OBAF in Regulated Environments

- > This guidance exists to help reformers apply outcome-based thinking in **non-ideal contexts**, not
- > to justify legacy approaches.

This complementary checklist is intended for teams operating in highly regulated environments that require adherence to contracting constraints, phase-gate processes, or sequential development cycles such as the V-model. Although these conditions are not ideal for agile practices, the checklist helps incorporate outcome-based agile thinking within these limitations. It also assists in identifying where existing constraints support or conflict with OBAF principles, encouraging a gradual shift toward more outcome-focused contracting philosophies and practices.

1. Frame Regulatory Needs as Immutable Constraints

- [] Have all mandatory regulations, standards, or audit requirements been clearly identified and documented as *non-negotiable constraints*?
- [] Are these constraints understood by the team as guardrails, not deliverables?

2. Treat Compliance Evidence as Outcomes

- [] Is the evidence of compliance (e.g., traceability, testing coverage, documentation) framed as part of the outcome definition?
- [] Are compliance deliverables produced iteratively and integrated into the workflow, not left to the end?

3. Enable Discovery Within the Boundaries

- [] Have areas where flexibility is *still possible* (e.g., usability, automation, user workflow) been clearly defined to allow innovation?
- [] Are hypotheses and solution exploration encouraged within the compliance envelope?

4. Automate Traceability and Documentation

- [] Are tools or lightweight methods in place to automatically capture decision logs, test coverage, change history, or requirement links?
- [] Are compliance artifacts continuously validated instead of batch-generated?

5. Engage Risk and Compliance as Learning Partners

- [] Are compliance/risk officers included early in framing outcomes and reviewing experiments?
- [] Is the compliance team encouraged to collaborate in defining what *evidence* of safety or control looks like—rather than dictating process?

Summary

Outcome-Based Agile is not a methodology; it is a recommitment to what agility was always meant to be: adaptive, user-centered, and value-focused. It is a call to drop the illusion of control in favor of the pursuit of clarity, progress, and real-world impact.

Outcomes, not outputs. Always.

Signatories

- Michel Blomgren mike@pkt.systems (2025-04-26)