

Power Platform Modernization

Playbook Part 2: Questions Leaders Should Ask & the Role of the Program Manager

A practical guide for business and IT leaders navigating Power Platform modernization — covering the critical questions to ask before work begins, the program manager's core responsibilities, and how to govern AI-assisted delivery responsibly.

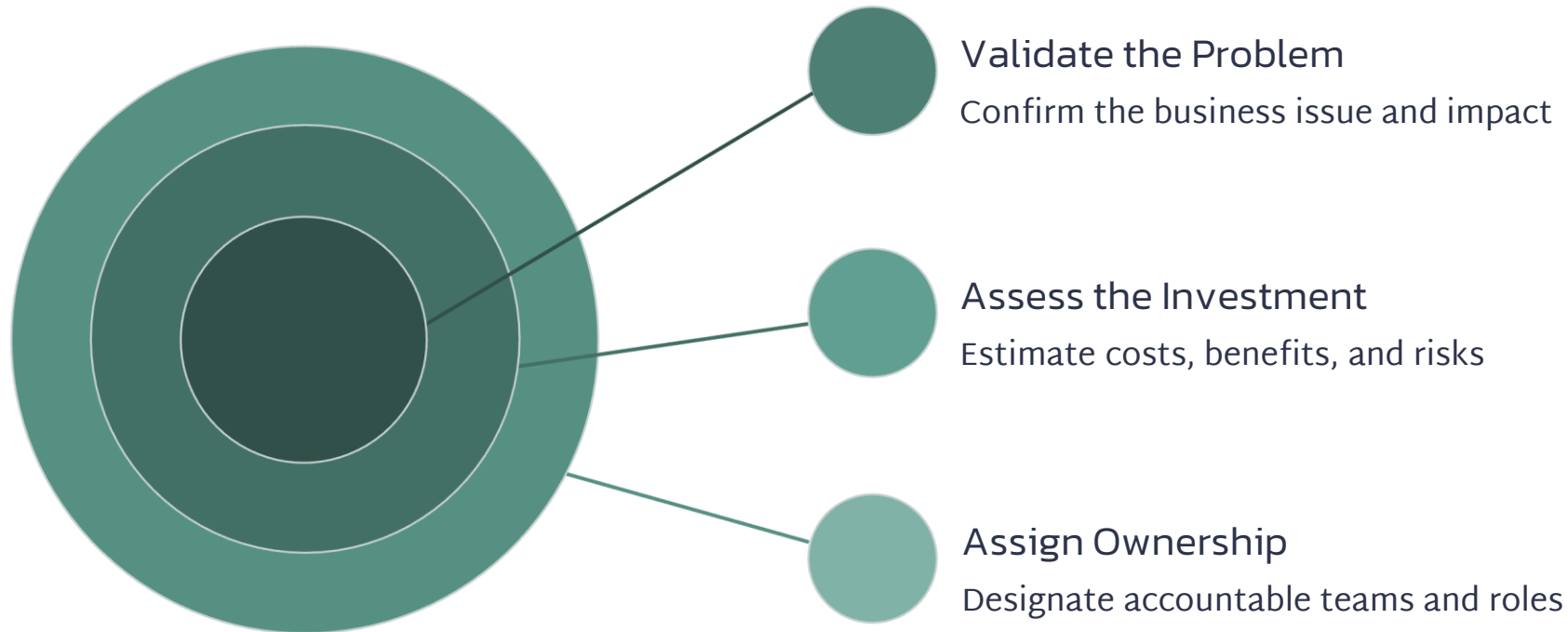
By Kimberly Wiethoff, MBA, PMP, PMI-ACP

#PowerPlatform #PowerApps #PowerAutomate #ApplicationModernization #DigitalTransformation #AI
#AIGovernance #ProgramManagement #ProjectManagement #PMO #ChangeManagement
#BusinessTransformation #AgileDelivery #ManagingProjectsTheAgileWay



Questions Leaders Should Ask Before Modernization Begins

Before approving a Power Platform modernization initiative, leaders must establish a shared understanding of the business problem, the investment required, and who owns key operating responsibilities. The right questions surface hidden assumptions early — before those assumptions become costly discoveries during delivery.



What Business Problem Are We Solving?

Every modernization initiative should begin with a clear, specific statement of the limitation driving the work. Leaders should press teams to name it precisely.

Slow Performance

Is the existing application creating bottlenecks that slow critical work processes or delay decisions?

Data Errors

Are manual workarounds or system gaps producing data quality problems that affect trust or compliance?

Rising Support Costs

Is maintenance overhead increasing faster than business value delivered by the current platform?

Blocked Change

Is the system preventing needed enhancements that the business requires to grow or adapt?

- ❏ Define the specific outcome that would justify the investment — not just the pain point, but the measurable improvement expected after delivery.

What Should We Improve or Retire Before Rebuilding?

Modernization is not just a technical exercise — it is an opportunity to question whether every existing step still serves a business purpose. Rebuilding a flawed process in a modern tool does not fix the flaw; it entrenches it in newer code.

Leaders should ask teams to review existing workflows and approval chains before a single requirement is written. Steps that exist because "we've always done it that way" should be challenged, simplified, or eliminated before becoming part of the new design.

 Recreating unnecessary steps carries years of accumulated inefficiency directly into the new application — at the full cost of a modernization project.

Questions to Ask the Team

- Which approvals are legally or contractually required?
- Which steps exist only because the old system required them?
- Where do users most commonly introduce errors or workarounds?
- What would a newly designed process look like if built from scratch today?

What Evidence Supports Power Platform as the Right Choice?

Platform selection should be supported by demonstrated evidence, not assumptions. Leaders should require teams to prove fitness before committing the full investment.



Prove High-Risk Requirements

Ask the team to demonstrate the most demanding or uncertain requirements using representative workloads — not simplified demos that avoid the hard cases.



Validate Integrations

Confirm that all required connections to data sources, external systems, and APIs have been tested under realistic conditions, not just confirmed as theoretically possible.



Review Licensing Assumptions

Ensure that per-user, per-app, and premium connector licensing costs have been modeled accurately for the actual user population and usage patterns.



Confirm Technical Review

Require sign-off from security, architecture, and data teams — not just the delivery team that has an interest in moving forward.

Do We Understand the Full Cost of Ownership?

Project budgets routinely undercount the true cost of a modernized application. Leaders should require a financial model that extends well beyond go-live.

Migration

Data cleansing, transformation, validation, and parallel-run costs during cutover

Licensing

Recurring per-user, per-app, and premium connector fees at full production scale

Training

Initial onboarding, refreshers for process changes, and onboarding of future staff

Support

Incident resolution, environment management, and break-fix coverage post-launch

Enhancements

Planned and unplanned improvements required as the business evolves after delivery

- ❏ Critically: identify who will fund the application **after** project funding ends. Unassigned post-launch funding is one of the most common causes of application decay.

Who Owns the Process, Data, and Release Decisions?

Name the People, Not Just the Roles

Ownership defined only at the team or department level creates ambiguity when decisions are contested or time-sensitive. Modernization initiatives require named individuals who hold specific authorities and can be held accountable.

What Must Be Assigned

- **Requirements approval:** Who has final authority to accept or reject scope changes?
- **Data quality:** Who is empowered to resolve conflicts or gaps in source data?
- **Risk acceptance:** Who can formally accept known delivery or operational risks?
- **Launch authorization:** Who gives the final go-ahead for production release?
- **Escalation path:** How will the team resolve disagreements when owners conflict?

Undefined ownership does not eliminate the decisions — it delays them until the cost of delay is highest.

Where Will AI Help, and Who Will Validate Its Outputs?

1

Define Approved Uses

Identify which tasks AI may assist with — documentation analysis, requirements drafting, formula generation, test cases — and which are out of scope

2

Set Data Sharing Boundaries

Specify what information teams are permitted to share with AI tools. Sensitive business data, PII, and confidential system details require explicit policy guidance

3

Assign Qualified Reviewers

Name the individuals responsible for validating each category of AI output — requirements, formulas, data mappings, and test cases — before acceptance

- ❏ AI accelerates drafting. It does not replace the judgment of the person who signs off on the result. Reviewer capacity must be planned, not assumed.

What Must Be True Before We Launch and Retire the Legacy System?

A go-live decision without agreed criteria is a judgment call made under pressure. Leaders should require measurable release gates before the project begins — not after the team is ready to ship.

1

Release Criteria

Define measurable, agreed conditions the application must meet before production launch — covering functional completeness, data accuracy, performance, and user acceptance

2

Rollback Triggers

Establish specific conditions that would require reverting to the legacy system, and verify that rollback procedures have been tested — not just documented

3

Archival Requirements

Confirm that historical data from the retiring system will be accessible, compliant with retention policies, and retrievable by support teams after decommission

4

Support Independence

Require the support team to demonstrate they can resolve common incidents without project team assistance before the legacy system is retired

The Role of the Program Manager

The program manager is the connective tissue of a modernization initiative. Their job is not simply to track tasks — it is to ensure that decisions happen at the right time, that evidence supports each transition, and that the effort stays connected to the business outcomes that justified the investment.



Translate Goals & Coordinate Dependencies

Translate Business Goals Into Measurable Outcomes

Work with the sponsor and process owners to establish baselines and targets before delivery begins. Connect every item in the delivery backlog to a business result — shorter processing times, fewer manual corrections, or more reliable access to accurate data.

Measurement does not stop at launch. Continue tracking outcomes through hypercare and the transition period so leaders can confirm the application is delivering the expected benefit — and act quickly if it is not.

Coordinate Dependencies Across Workstreams

Application development depends on data readiness, integration availability, environment configuration, security decisions, and sustained user participation. None of these happens automatically.

The program manager maintains a clear, current view of every dependency — with a named owner and a required completion date. Blocked decisions are surfaced early, with a specific explanation of their effect on the release plan. Waiting to report a blocked dependency is not an option.

Establish Decision Rights & Govern AI-Assisted Delivery

Decision Rights & Acceptance Gates

Define who approves scope, architecture, data reconciliation, business acceptance, and production launch. Each phase requires an agreed deliverable and explicit acceptance criteria — not a general sense that the work looks finished.

Unresolved issues go to the appropriate decision-maker with options and stated consequences. Approved exceptions are recorded with the responsibilities attached to them. Verbal agreements are not sufficient.

AI-Assisted Delivery Governance

AI use must follow the organization's approved data-handling rules and review standards. Each AI-assisted deliverable has a named owner responsible for its accuracy and formal acceptance — not just the person who generated the output.

Account for review effort in the delivery plan. Faster drafting only improves delivery speed when the team has the capacity to validate what the AI produced. Unreviewed output is not a completed deliverable.

Lead Adoption & Maintain Accountability Through Benefits Realization

Lead Adoption and Operational Readiness

User involvement, training, and support preparation must be planned alongside development — not added at the end when the schedule is compressed. Use feedback from demonstrations and structured testing to identify adoption barriers before they surface in production.

Operational readiness is not a checklist. Support teams must have the access, documentation, and hands-on experience needed to resolve incidents without assistance from the project team.

Sustain Accountability Through Benefits Realization

Track stability, adoption rates, and business outcomes throughout hypercare and the agreed transition period. Unresolved issues and planned future improvements require named owners — not open items on a shared log.

The program manager's work is complete when the organization has accepted a sustainable capability, demonstrated operational readiness, and established clear ownership for measuring its continuing value.

Keep AI Assistance Reviewable

The strongest starting points for AI assistance in this playbook are documentation analysis, requirements drafting, development support, and test-case generation. Each produces output that a qualified person can inspect, challenge, and formally accept.

Apply Consistent Quality Standards

AI-assisted work is held to the same quality bar as any other deliverable. Review generated content against source material, test generated logic in a representative environment, and validate proposed data mappings through formal reconciliation.

Named Owners Approve Release Decisions

Regardless of how supporting work was prepared, release decisions require a named human owner to formally approve acceptance. The method of production does not change the accountability standard.

The PMO Checkpoint

A single question keeps AI governance practical: **What did AI help produce, who reviewed it, and what evidence supports its acceptance?** That question keeps delivery focused on a working business capability rather than a volume of generated artifacts.

- ❏ The project succeeds when employees can complete their work, the organization can trust its data, and support teams can sustain the application after the project team has left.