Part 4 of 20

Requirements Definition: What the Business Actually Needs

The most expensive mistake in ERP implementation is inadequate requirements definition — here is the rigorous process for avoiding it

CONTENTS OF THIS PART

1. What You Will Learn and Why It Matters
2. The Requirements Gathering Process
3. Functional Requirements: The Finance Capabilities the System Must Support
4. Technical Requirements: Integration, Security, and Scalability
5. Future-State Process Design
6. The Requirements Traceability Matrix
7. Actions to Take in the Next Thirty Days

WHAT YOU WILL LEARN AND WHY IT MATTERS

Inadequate requirements definition is the most expensive mistake in ERP implementation — not because it causes the largest single failure, but because it causes the most pervasive and most persistent failure mode: the delivered system that technically works but does not deliver the analytical capability the organization needed. An ERP implemented without rigorous requirements is an ERP configured to the vendor's default design rather than to the organization's actual business model, and the gap between default design and business need produces the manual workarounds, reporting limitations, and analytical constraints that were supposed to be eliminated by the new system.

This part covers the complete requirements definition process: who to involve, what to ask, how to document requirements with the precision required for vendor evaluation and implementation governance, the difference between current state documentation and future state design, and the requirements traceability matrix that prevents scope creep from eroding the analytical ambitions of the implementation. Requirements definition is not glamorous work — it is detailed, time-consuming, and organizationally demanding. It is also the activity that most determines whether the ERP implementation delivers on its promise.

THE REQUIREMENTS GATHERING PROCESS

Requirements gathering is a structured discovery process that identifies the full range of capabilities the new financial system must provide — not just the capabilities the current system provides, but the capabilities the business needs and does not currently have. The distinction is critical: requirements gathered primarily from the current system's functionality will produce a new system that replicates the current system's design with better technology. Requirements gathered from the business's future analytical needs will produce a new system that enables the financial intelligence platform.

The requirements gathering process should involve four stakeholder groups. The first group is the finance leadership team — the CFO, the controller, the VP of FP&A;, and the senior finance managers who are responsible for the processes the new system will support. Finance leadership provides the strategic requirements: the analytical capabilities the finance function needs to fulfill its mandate of translating strategy into numbers and enabling better decisions. These strategic requirements define the reporting architecture, the chart of accounts design, and the integration priorities.

The second stakeholder group is the finance operations team — the accounts payable, accounts receivable, payroll, and general accounting professionals who perform the transactional work the ERP will automate. This group provides the process requirements: the specific workflow capabilities, automation features, and control mechanisms the system must support to enable efficient, accurate financial operations. These process requirements define the module configuration, the approval workflow design, and the automation priorities.

The third stakeholder group is the business partner population — the sales, marketing, product, and operations leaders who depend on financial reporting and who interact with the ERP through procurement, expense reporting, and project accounting processes. This group provides the business user requirements: the reports they need, the data entry processes they will perform, and the self-service analytics capabilities that would reduce their dependence on the finance team for routine data requests.

The fourth stakeholder group is the IT and systems team — the technology professionals who will manage the ERP's technical infrastructure, perform integrations, and support the system post-go-live. This group provides the technical requirements: the security and access control requirements, the integration architecture requirements, the performance and scalability requirements, and the disaster recovery and business continuity requirements.

FUNCTIONAL REQUIREMENTS: THE FINANCE CAPABILITIES THE SYSTEM MUST SUPPORT

Functional requirements describe the specific finance capabilities the new system must provide — the processes it must support, the reports it must produce, and the analytical questions it must be able to answer. They are the core of the requirements definition and the primary basis for vendor evaluation.

The chart of accounts is the most consequential single functional requirement in ERP design. It is the dimensional framework that determines what analytical questions the system can answer directly and what questions require manual analysis. A chart of accounts designed with only the dimensions required for statutory financial reporting — legal entity, cost center, natural account — will support basic management reporting but will not support the analytical questions the FP&A; team needs to answer about product profitability, customer profitability, or project-level economics. A chart of accounts designed with additional dimensions — product, customer segment, channel, project, geography — enables significantly richer reporting directly from the system.

The multi-entity and consolidation requirements deserve particular attention for any company with multiple legal entities, subsidiaries, or geographic operations. Multi-entity requirements specify how intercompany transactions are recorded and eliminated, how currencies are converted at the appropriate rates for financial statement purposes, and how consolidation reports are produced that aggregate the financial results of multiple entities. For many growth-stage companies, the inadequacy of their current system's multi-entity capabilities is the primary driver of the ERP replacement decision.

The revenue recognition requirements are among the most technically complex functional requirements for subscription software companies subject to ASC 606. The system must be able to handle variable consideration, contract modifications, standalone selling price allocation across performance obligations, and the distinction between revenue recognized over time and at a point in time. For companies with significant professional services revenue, usage-based pricing, or customer contracts that include both subscription and professional services elements, the revenue recognition requirements may be complex enough to justify a dedicated revenue recognition module as a best-of-breed component rather than

relying on the native ERP revenue module.

The reporting requirements should be documented at the level of specific management reports — the monthly income statement by segment, the customer profitability analysis, the headcount report by function and level — rather than at the level of general capabilities. Documenting specific reports forces both the requirements gathering team and the vendor to think concretely about whether the system can produce exactly what the organization needs, rather than whether it has general reporting capabilities.

TECHNICAL REQUIREMENTS: INTEGRATION, SECURITY, AND SCALABILITY

Technical requirements define the operational and architectural characteristics the system must possess to function reliably in the organization's technology environment. They are less visible than functional requirements but equally important: a system with excellent functional capabilities that has inadequate security, cannot integrate with critical operational systems, or degrades in performance at peak transaction volumes will fail to deliver its promised value regardless of the quality of its financial modules.

Integration requirements specify every system connection the ERP must support, the data that flows in each direction, and the technical approach to each connection. For a typical growth-stage technology company, the integration requirements include connections to the CRM for customer and contract data, the HCM for employee and compensation data, the billing and subscription management platform for subscription revenue data, the payroll platform for payroll transaction data, the expense management system for expense report data, and the FP&A; planning tool for plan and actual data exchange. Each integration requirement should specify the frequency of data exchange — real-time, hourly, daily — the data volume at peak load, and the error handling requirements for failed integrations.

Security and access control requirements define how user access to financial data must be managed. Financial systems are among the most security-sensitive applications in any organization — they contain the financial records that support regulatory compliance, the vendor banking information that creates fraud risk, and the management data that creates information security risk. The access control requirements should specify the principle of least privilege — users should have access only to the data and functions required for their role — the segregation of duties controls required to prevent fraud — the same user should not be able to both create and approve a vendor payment — and the audit trail requirements that support both internal control testing and external audit.

Scalability requirements define the system's performance expectations at current and anticipated transaction volumes. An ERP that performs adequately today but degrades significantly at three times the current transaction volume will become a constraint on growth at precisely the moment when reliable financial systems are most critical. Scalability requirements should specify the expected transaction volumes at current scale and at the anticipated scale at the end of the implementation's useful life — typically five to seven years for a new ERP — and should require the vendor to demonstrate performance at the anticipated scale through load testing or through reference customers operating at that scale.

FUTURE-STATE PROCESS DESIGN

One of the most important and most frequently missed opportunities in ERP requirements definition is future-state process design — the deliberate design of the business processes the new system will support, rather than the documentation of the current processes the new system will replicate.

The distinction between documenting current processes and designing future processes is fundamental. Current process documentation captures how the finance function works today, including all the manual workarounds, approval bypasses, and compensating controls that have accumulated over time. An ERP configured to support these current processes will automate many of the manual steps, which is valuable, but will also perpetuate the process inefficiencies and control weaknesses that have developed in the legacy system environment.

Future-state process design starts from first principles: what is the optimal way to perform each financial process given the capabilities of the new system? Which approval steps exist because the current system cannot enforce controls automatically, and could therefore be eliminated with proper system configuration? Which manual reconciliations are required because data does not flow automatically between systems, and could be replaced by automated reconciliation in a well-integrated environment? Which reports are produced on a manual basis because the current system cannot produce them automatically, and could be replaced by system-generated reports in a modern ERP?

The future-state design process requires the participation of a finance professional who understands both the business requirements and the capabilities of the target system — typically either a senior finance manager with ERP implementation experience or an implementation consultant who can bridge the gap between finance requirements and system capabilities. The output of the future-state design is not a list of requirements but a set of process flows — visual diagrams and written descriptions of how each financial process will work in the new system environment — that serve as the specification for the system configuration and the basis for user training.

THE REQUIREMENTS TRACEABILITY MATRIX

The requirements traceability matrix is the analytical document that connects every requirement to the system design decision that addresses it, and that provides the governance mechanism for managing scope throughout the implementation. It is the single most important governance tool for preventing the scope creep that is one of the primary causes of ERP implementation overrun.

The requirements traceability matrix is a comprehensive table with rows for each documented requirement and columns that show how each requirement is addressed in the system design. The columns typically include: the requirement identifier and description, the priority classification — must-have, should-have, or nice-to-have — the system design element that addresses the requirement — the specific module,

configuration, or integration that provides the capability — the implementation phase in which the requirement will be addressed, and the test case that will verify that the requirement has been correctly implemented.

The must-have versus should-have versus nice-to-have classification is the most important element of the matrix because it creates the basis for scope management throughout the implementation. Must-have requirements are those whose absence would cause the implementation to fail to deliver its primary business objectives — the system cannot go live without them. Should-have requirements are those that would significantly improve the value delivered but whose absence would not prevent go-live. Nice-to-have requirements are those that would be beneficial but that are not essential to the primary value proposition.

When the implementation encounters budget or timeline pressure — as most implementations do — the must-have versus should-have classification provides the analytical framework for scope decisions. Nice-to-have requirements can be deferred to a post-go-live phase without compromising the business case. Should-have requirements can be deferred if necessary, with explicit acknowledgment of the value impact. Must-have requirements cannot be deferred without fundamentally changing the nature of the implementation, and should trigger either additional resource investment or a formal business case revision if they cannot be addressed within the current scope.

The requirements traceability matrix also serves as the acceptance criteria framework for the implementation. At go-live, every must-have requirement should be verifiable against the specific test case in the matrix. The go-live readiness assessment — the decision about whether the system is ready to replace the legacy system in production — should be based at least in part on the percentage of must-have requirements that have been successfully tested and verified.

ACTIONS TO TAKE IN THE NEXT THIRTY DAYS

The following actions will begin the requirements definition process and produce the initial analytical foundation for any ERP initiative.

The first action is to conduct stakeholder interviews with each of the four groups described in this part — finance leadership, finance operations, business partners, and IT — using a structured questionnaire that covers functional requirements, technical requirements, and future-state process aspirations. Document the results systematically and identify the requirements that appear consistently across multiple stakeholder groups — these are the highest-priority requirements for the initial requirements document.

The second action is to develop an initial list of the fifteen to twenty must-have functional requirements — the capabilities the new system must provide for the implementation to deliver its primary business objectives. For each must-have requirement, document the specific current limitation it addresses, the specific future capability it represents, and the specific business outcome it enables. This must-have list

will be the most important input to the vendor evaluation process in Part Five.

The third action is to document the current chart of accounts and assess its analytical adequacy against the reporting requirements of the business. Can the current chart of accounts support direct reporting at the product, customer segment, channel, and project levels that the FP&A; team needs? Which dimensions are missing? What would the future-state chart of accounts need to include to support the analytical reporting the business requires? This assessment will produce the chart of accounts requirements that are among the most technically specific and most consequential requirements in the entire implementation.

The fourth action is to develop initial process flow diagrams for the three to five financial processes that will most significantly change in the new ERP environment — typically the financial close process, the accounts payable process, and the order-to-cash process. For each process, document both the current state and the desired future state, identifying the specific changes that the new system enables and the organizational behavior changes that will be required to operate the new process effectively. These process flow diagrams will form the foundation of the implementation design work in Part Nine.

CLOSING PERSPECTIVE

Requirements definition is the investment in analytical clarity that prevents the most common and most costly form of ERP failure: the implemented system that technically works but does not deliver the financial intelligence platform the organization needed. That failure mode is not recoverable within the project — by the time it is visible, the budget has been spent and the configuration has been completed. It can only be prevented by doing the analytical work before the implementation begins.

The time invested in rigorous requirements definition — the stakeholder interviews, the future-state process design, the chart of accounts analysis, and the requirements traceability matrix — is time that pays back many times over in the quality of the system that results. It is also time that produces the analytical clarity required to evaluate vendors effectively, which is the subject of Part Five.

COMING NEXT IN THE SERIES

Part 5 — Vendor Selection: The Analytical Framework for Choosing Right

Part Five covers the full vendor selection process — the weighted evaluation framework, the scripted demonstration process, the reference check discipline, pricing model analysis, and contract negotiation. It is the analytical counterpart to the requirements definition process, ensuring that the vendor selected is chosen because they best meet the documented requirements rather than because they gave the most compelling demo.