

ERP System Implementation Project Planning

Successful ERP implementations are, by seemingly trite definition, simply those that don't fail; but this *isn't* trite, because failure is so very common. Failure follows directly from allowing a few well-known risks to go unmanaged. They go unmanaged because control of *results* slips from buyer to seller. The buyers (the user company) are *the only people in the world* capable of seeing the new system as a tool for running their particular business. The sellers can't and don't think in terms of their client's business; they are hired because they think in the technical terms of their product. Control usually passes from buyer to seller at the very outset, without anyone even noticing, simply because no one sees a practical alternative. Buyers don't see how they can get and keep control of results while still getting what they need from providers, and sellers don't see how they can give up this control while protecting their contractual interests. The loss of control can be avoided if some basic principles of project management are rigorously applied from the inception of the acquisition and implementation process, and well before what the *seller* calls the *implementation project* begins.

In the preceding topics we discussed two important themes:

- In "ERP System Acquisition & Implementation Viewed as a Single Project" we discussed how a properly functioning ERP system is a kind of highly selective mirror of a company's operations. We discussed what has to happen prior to implementation if the mirror's reflection of the company is to be accurate, complete and apt.
- In "ERP System Acquisition Project Planning" we discussed the immense complexity of ERP software and the correspondingly difficult task implementers face in selecting just the right combination of configuration options that best describes the user company and its operations.

The payoff for pre-implementation planning (or the payback for its absence!) comes in the implementation phase itself. Even if the implementation team sets out from the very start to configure their product by working to a comprehensive blueprint of the desired system, some time will pass before results can actually be seen. During this period the team's progress can (and should) be monitored in various ways (using established project management techniques), but the actual results of their work – which is to say, of the project as a whole to date – cannot be seen yet. It is only when the first fruits of their efforts can be viewed by the user community that anyone can really begin to evaluate the success of their work.

How does this matter? How, especially, does it matter in terms of risk management?

Let's oversimplify and assume that the results of the implementation team's work appear continuously throughout the project. (This isn't really true. Results usually become visible in bunches, and some results – those that depend on the integration of many individual modules – only appear toward the end. But let's simplify for the sake of example.) The schedule calls for an implementation phase of W weeks. As the implementation emerges, an unknown number of problems will be revealed; call this number P . In our overly simple model, the appearance of problems will be dispersed evenly throughout the continuous

emergence of the configured product. This means that after $\frac{1}{2} W$ weeks have past, $\frac{1}{2} P$ problems will have emerged – and fully $\frac{1}{2}$ won't have shown themselves yet (we remind you again: this model errs widely on the side of optimism).

Problems have to be dealt with as they emerge. In some cases the project team can correct them in stride. In other cases, the team will have to backtrack and redo work previously thought complete. In particular, the more tests that have occurred (and been deemed complete) when any problem is discovered and addressed, the more tests that will likely have to be repeated; and testing comprises (or should comprise) a large portion of the implementation effort.

Minimizing P will obviously have a positive impact on the project's success, and every effort should be made to do that. But P can never be reduced to zero, and the full magnitude of P cannot be known until the final run of acceptance testing has been passed with every known problem resolved (and shown to be resolved). Thus **two goals** emerge that are both at least as important as the admittedly important goal of minimizing P :

- Given any possible value for P , the earlier in the project that the full extent of P is known, the more efficient and compact the project will be; restated negatively, the longer it takes to discover the total extent of P , the more work the implementation team will have to perform, and the greater the stress on the project schedule (and on schedule-driven costs). **Therefore the early discovery of problems is a vital objective.**
- When any problem is discovered, its magnitude depends not only on the work required to fix it, but also on the potentially greater impact that fix has on other work previously thought complete. **Therefore testing should be planned so as to limit and localize the impact of any problems revealed by the tests.**

These things don't happen without a very explicit kind of planning that should be complete before what is often called "implementation" is even begun. This kind of planning covers four general areas:

- Predefining the expected results of the implementation in ways that the implementation team can work to, so that disconnects will appear before (or at least during) the initial process of configuration and not as revelations deferred to some far less convenient stage in the project.
- Stating the expected results of the implementation in terms of intermediate, data-driven goals, so that the implementation team can anticipate the global effects of local configuration decisions without waiting for full integration testing (i.e., for the **scheduled** end of the project).
- Defining the entire testing process so as to localize the impact of those problems that inevitably do emerge.
- Planning the entire implementation so as to monitor and control these processes in accordance with the predefined goals of the new system.

This is a high level description of what we deliver.