Seilevel Whitepaper

Gathering Requirements for Migration Projects

By Joy Beatty, Director of Services

Seilevel
requirements defined

Many IT projects are focused on migrating existing functionality to a new system. The goal of the project may be to move to a new platform, build a new system from the ground up with the exact same functionality, or to move a piece of functionality from one system to another. Justifications for such projects include moving off an unsupported technology, gaining performance improvements or integrating with other enterprise systems.

The requirements gathering effort for migration projects is notably different than for a new system being built from scratch or for adding new functionality to an existing system. At a high level, the distinctions are in scope definition, understanding original business needs, working with end users, discovering the end-to-end functionality and IT involvement.

Scope

As with any project, the scope definition on a migration project is critical. The variance in this type of project comes from the specific considerations in scope. Scope discussions should take into account whether to include the following:

- *migration of all functionality*
- *phasing pieces of functionality over a rollout timeframe*
- *adding or changing functionality*
- *updating user interfaces*

Prior to the requirements phase, a decision should be made regarding whether all functionality is being migrated immediately. If only a piece of a system's functionality is being migrated, a clear boundary must be drawn around what is in scope for the project. As an example, for one project, inventory management comprised about 10% of the total functionality in the existing system. This piece was being moved to a new system that was only going to house the inventory functionality. The original system stayed in place, and the inventory management piece was turned off in it. This defined a very clear boundary for the requirements gathering effort.

A joint business and IT decision must be made when considering if any new functionality will be developed. Often, migration projects are driven by IT needs, but the business cannot wait for a new system to be completed before getting additional functionality. The trade-offs have to be weighed, because not adding new functionality may simplify the project allowing an earlier launch.

When working with systems that have user interfaces, there is a choice to whether the changes will be limited to bug fixes, simple style changes or a complete overhaul. This decision should take into account the users' needs. If you are working with a call center application where minimizing total call time is crucial and the sales reps are skilled in the existing user interface, changes to the interface should be minimal to avoid impacting the reps' learned efficiencies in the current user interface.

Understanding business needs

If there is a detailed requirements specification for the original system, a valid question to entertain is whether you can just reuse that specification to build the new system. In theory the answer is "yes", but in reality, that can be a risky path to take. Even though you are replicating the existing system, you still need to ensure that you are not redeveloping functionality that was incorrectly defined in the first system. To avoid this, start with the original specification, and then interview users to establish how they use the actual functionality.

On the other hand, there may be a temptation to ignore the original specification, even as much as to ignore the existing system, and build the requirements from the ground up. This would require revisiting the true business needs and developing new requirements. If you rebuild the requirements from the ground-up, you will spend more time in gathering and writing the requirements and development will spend more time building them. It actually may be more cost efficient to build what you have and modify it in future releases. In fact, with this approach, it really is no longer a migration project and instead is just building a new system with new functionality to support existing business needs.

Each project will be unique regarding which side of the spectrum is most appropriate. However, somewhere in the middle of these two, there is an approach that uses elements of both to get to the best system in the most cost effective manner.

Working with end users

As with any requirements gathering effort, you still need to utilize the end users, but in migration projects, there are differences in how you work with them.

In developing new functionality, it is helpful to work with users at a higher-task level to grasp the goals of their activities. And while that view of their role is still essential, it is also important to understand how they actually do the tasks in the existing system. The difference in migration projects is that significantly more of the overall requirements effort will be spent understanding the current system usage.

Similarly, with new functionality, it is often appropriate to discourage the user from focusing on user interface design. While in a migration project, it makes perfect sense to focus on the existing user interface. As an example, the sales rep in the call center application can describe the generic steps to find a product in the catalog, add it to a cart, capture the customer's information and checkout. However, it is more useful to actually observe the sales rep doing this task and capture exactly what the screen choices are.

If the goal is explicitly to just migrate functionality, you should still ask the users "What do you need to be able to accomplish in the system?" However, there is a much greater focus on "How do you use the system you have to do it?" and very little focus on "What do you want the system to do that it doesn't already?" The unfortunate news here is that the users may not see how this type of effort will benefit them, so it is important that they understand the business objectives behind the project.

Discovering the end-to-end functionality

As with any project, it is important to capture the entire set of functionality. One of the great advantages of a migration project is that you do have an existing system to work from, including user and system interfaces. For user interfaces, the menus can be used to decompose the functionality. The existing screens can be clicked through to discover the alternate paths through the system. For system interfaces, you can get a complete list of the interfaces and what data is passed between them to ensure they are covered in the requirements.

In meeting with the end users to understand how they use the system, you can easily capture use cases. With new functionality, use cases are a crucial model to discovering all the functional requirements, though that advantage is less notable in migration projects. The use cases are still valuable to give the full picture of how the functionality fits together to accomplish user goals, and they provide an organizational structure for the requirements. The significant difference in writing the use cases is that there should be more reliance on using the existing user interface design.

In the case where no one knows exactly how something works, there is a convenient option to just look at the code and the database schema to understand the actual implemented business rules. In fact, theoretically, the development team could just use the original code to develop the new system. Though realistically, this is dull and error-prone for a developer.

Working with IT

In a migration project, it should be expected that there is far greater involvement from IT in the requirements phase of the project. In discovering new functionality needs, IT may not be fully engaged early in the project to avoid stifling the creativity of the business users. However in a migration project, it is crucial to engage them as early as possible, and there is absolutely no risk to that involvement.

The development team can help with a technical decomposition of the system functionality. They will have a clear view of the data elements and ensuring those are all well understood in the existing system. For example, in the migration of a financial system, there are many underlying matrices of data and business rules. The business users cannot recall the all of specific detailed business rules around those matrices. It would be far more efficient for a developer to look at the existing code to help understand the specific matrices and the associated behavior.

One of the great risk mitigations for a migration project is that the development team will be looking at the code of the existing system and therefore it can serve as one last checkpoint to ensure functionality is not overlooked.

Discovering what is not used

The scope of a migration project should include identifying functionality that is not used, so that the development team does not waste cycles building it. This is precisely why defining the requirements is not as simple as just reusing the original requirements specification or having the development

team build it straight from the existing system code. A suspect list of unused functionality will be discovered in the user interviews. For example, if users are asked “What happens if you select this option on the screen?” and there are consistent answers like “I don’t know, I never do that,” then this is functionality to be flagged for removal. In an ideal world, IT may be have data on what screens are never accessed or what data elements never are updated in the existing system.

It may be difficult for the business to sign off on saying that a piece of functionality is not used. However, that is exactly what is necessary. Once there is that list of target functionality to be removed in the new system, every single one of the identified user groups of the system must be interviewed to confirm the list.

Resource skill set

Any great product manager can identify the requirements for a migration project. However, this is one type of project where a product manager with a technical background can be a great advantage. Such a product manager will have the capability to actually read the code and architecture diagrams if required to gather the requirements.

In the end, certainly there are many similarities in the requirements efforts of migration projects and new functionality projects. However, there are some important differences, and it is critical to the project’s success that these variations are recognized before starting the requirements phase of the project.

About Seilevel

Founded in Austin in 2000, Seilevel is a professional services firm that creates requirements for Fortune 1000 customers. The Seilevel mission is simple: redefine the way companies create software requirements. Today, only a small percent of IT projects succeed; the primary reasons for project failure are poorly defined or missed requirements. Leading companies turn to us to define their requirements because of a proven approach to software requirement definition that saves development dollars and maximizes resources. Seilevel gets the requirements right, so our clients get their software right.

About the author

Joy Beatty has deep knowledge of all aspects of the software development life cycle, particularly with online and offline sales applications, pricing applications and e-commerce solutions. She has been instrumental in developing a strong methodology at Seilevel for use case development and requirements gathering.

For more information, visit us at www.seilevel.com