White Paper

Managing TM1 Projects

What You'll Learn in This White Paper:

- Traditional approaches to project management
- A more agile approach
- Prototyping
- Achieving the ideal outcome
- Assessing project teams
- Managing large TM1 projects
- Keys to successful TM1 projects

TM1 is an extremely flexible application development platform and includes all of the tools required to develop a complete application from end to end. The flexibility of the environment allows applications to be developed very quickly; application prototypes can be developed in a matter of hours and full applications can be developed in just days or weeks.

The TM1 development environment is also very fast to learn with newcomers to TM1 capable of developing their own applications within just a couple of days of training. The system is very intuitive, especially for those who have developed complex spreadsheet models in the past.

The flexibility described above enables applications to be built quickly and easily, however this level of flexibility can also have some undesirable side effects if projects are not managed well. It is very easy to create "bad" applications that may solve specific business problems, but leave little room for future development and improvement.

To ensure that the TM1 platform is fully leveraged and systems are developed to a high standard there are two key principles that are recommended:

- Implementation of systems that follow best practice development principles
- Management of TM1 projects using a methodology that complements the technology

This document is intended to address the second point; that is to describe an approach to managing TM1 projects that produce successful, high quality TM1 systems. The remainder of this white paper will describe the recommended project management approach in more detail and compare and contrast it with other project management approaches.

Project Management Approaches

There are many different approaches to managing the implementation of IT systems including:

- The Waterfall Approach
- The Prototyping Approach
- The Incremental Approach
- The Spiral Approach
- The Agile Approach

Within each of these approaches there are countless variations on the theme; however the approach remains generally consistent.

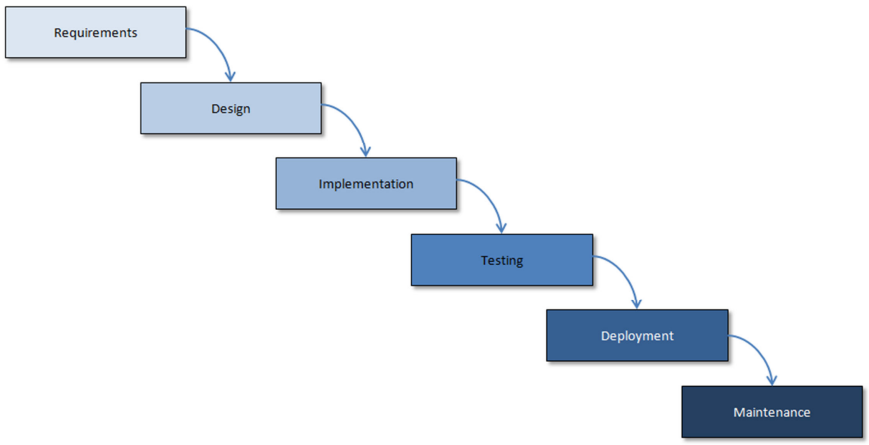
The most common approach to implementing IT systems is the Waterfall approach. The waterfall approach is described briefly in the next section.

The Waterfall Approach

Overview

The waterfall approach breaks a project down into discrete phases. Each phase is undertaken in a linear fashion and the completion of each phase leads directly to the commencement of the following phase. Each phase generally requires a form of sign-off from a number of parties involved with the project. Theoretically, the next stage of the project should not commence until the previous phase has been completed and signed off.

In practice, there are variations to the stages used in the waterfall approach but generally it follows a pattern of: requirements, design, implementation, testing, deployment and maintenance. The following diagram illustrates this approach:



It is common to use different groups of people to perform each phase as they require different skill sets. For example the requirements gathering phase is generally undertaken by a business analyst, the implementation phase is undertaken by application developers and so on.

As each stage needs to be completed before moving to the next stage, the requirements and design need to be decided early in the project before the implementation begins. The implementation needs to be complete before the testing begins and so on.

Advantages of the Waterfall Approach

The waterfall approach works well for large IT projects, particularly operational or transactional systems that have a lot of interrelated components. It also works well when projects have large teams where the work of one group of team members has direct impacts on other groups within the team.

Defining requirements and design up front in these situations helps ensure that everyone on the team has a clear understanding of what needs to be produced at the start of the project. The waterfall approach also closely follows the traditional software development life cycle that most IT professionals are familiar with.

Disadvantages of the Waterfall Approach

Due to the fixed phases of the waterfall approach the design of the system is effectively locked in very early in the project. If new information comes to light or there is a change in the business requirement after the design has been signed off, it is often difficult and costly to make changes.

The waterfall approach generally uses a fairly rigorous change management process. This can make adapting to changes mid stream difficult or impossible. In many cases the change management process becomes a deterrent to requesting changes at all. This may be effective in controlling time and cost, but does not encourage the best possible outcome for the system if people feel it is simply not worth the effort of raising a change request.

The waterfall approach requires a lot of up-front documentation. This documentation requires a lot of time and effort to produce and is done without seeing how the requirements will be implemented in practice. Often requirements documents become out of date quickly, especially if any major change requests are made during the implementation.

The sequential nature of the waterfall approach generally leads to longer project timelines. The design team cannot start work until the requirements gathering phase has been completed; the development team cannot start work on the implementation until the requirements and design are completed and so on.

Users generally don't get to see a functioning system until very late in the project lifecycle. If users don't get to see the system until the testing phase it leaves very little time to make changes to the

system before it is finalised. Changes that are made at this point are generally more expensive than what they would have been if identified and actioned earlier in the project.

The waterfall approach is very restrictive for flexible development platforms like TM1. Transactional systems that perform specific tasks might be better suited to the waterfall approach, but systems like TM1 are designed to allow the end user to choose how they use the system. Accordingly, TM1 systems require functionality that may not be known right at the start of the project. Users need time to be exposed to the technology and its capabilities as the system is evolving.

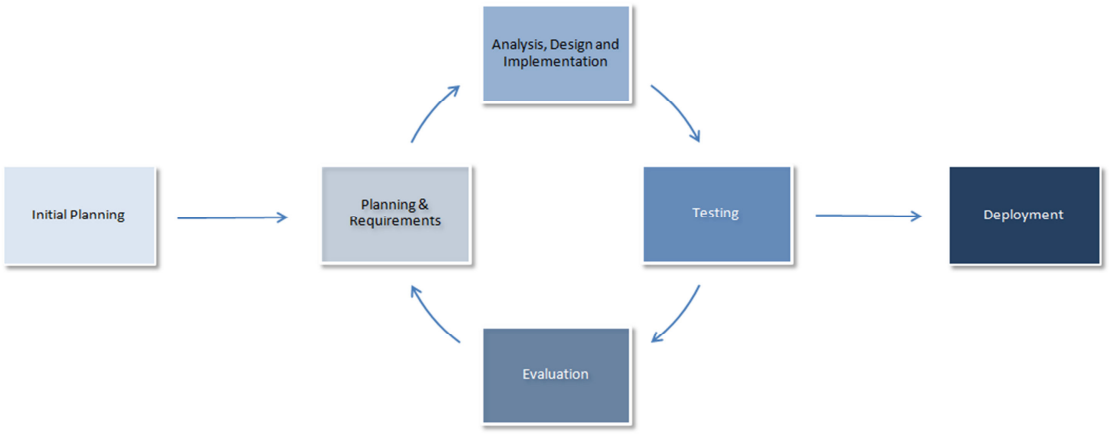
A More Agile Approach

Overview

Flexible systems like TM1 require a more agile approach to development. Users should be exposed to how the system works very early in the project lifecycle to allow them to understand the technology and grasp concepts such as multi dimensionality. This in turn results in more informed decision making when users are defining the requirements of the system which ultimately results in a system that best satisfies the business requirement.

In order to achieve the ideal system, it should be developed in a series of "iterations" which hone in on the ideal solution. Users and developers should work together to achieve this ideal outcome. Iterations can be as short as a few hours or may take several weeks. The important thing is to produce some tangible parts of the system that the user can experience and provide feedback on. This may be a simple prototype or a more sophisticated piece of the system. Based on this feedback, changes can be made during the next iteration and so on.

The following diagram illustrates this approach:



As the diagram shows, the first step using this more agile approach is initial planning. This phase should be very short and is used to understand the overall business requirement. Following the initial planning phase the project then moves in to the first iteration of the development of the

system. The first iteration may be a first attempt at the whole system or just one part of the system. Within the iteration, the planning & requirements phase will involve some more detailed understanding of the requirements of the current iteration, followed by analysis, design and implementation of the piece of the system being developed. During this time the end user should remain involved in the process providing valuable feedback to the developer as the system evolves. Following this there will be some testing of the functionality developed in this iteration followed by an evaluation of what has been produced.

Following the evaluation of the first iteration, there will be subsequent iterations (if necessary) until the system has achieved the desired outcome. At the conclusion of all of the iterations the project moves into the deployment phase.

Advantages of a More Agile Approach

The design of the system is not locked in at the start of the project the way that it is with the waterfall approach. Because the users and developers are working closely together and evaluating and re-evaluating the system during its development, it is more likely that the ideal solution will be found.

The system is adaptable to changes mid stream without major impact to the progress of the project. The flexibility of the TM1 platform enables system changes to be made quickly and easily.

The more agile approach is less dependent on up front documentation compared with the waterfall approach. By creating prototypes and early iterations of the system users can clearly see and help direct the actual system itself rather than just a documented version of what they might get.

This approach also involves the user throughout the lifecycle of the implementation, rather than just at the beginning and the end. Once again, this increases the likelihood of achieving the ideal solution by way of the continuous feedback loop between users and developers

Disadvantages of a More Agile Approach

The agile approach by its nature allows more flexibility to the design and development of the system. This of course has disadvantages as well.

The more agile approach can lead to "cowboy" behaviour if not correctly managed. Developers may feel they have free reign to do as they please given the less formal environment they are working in. It is up to the project manager to ensure that adequate progress is being made and each iteration leads to continuous improvement in the system.

As requirements are not as clearly defined up front compared to the waterfall approach there is a greater likelihood of scope creep with the more agile approach. Once again, it is the project manager's responsibility to ensure scope is being correctly managed.

It is harder to predict delivery dates using the more agile approach due to the iterative nature of the process.

Prototyping

The best way for a user to understand the proposed behaviour and functionality of a system is for them to see it and to use it. Obviously a user won't see the final system and all of its fully functional capabilities until the development is complete, but it is possible to demonstrate intended functionality and behaviour by prototyping it. Prototypes can assist greatly in demonstrating concepts and ideas.

A prototype is a working version of a system or part of a system that shows enough of the functionality and behaviour of a system so that users can make key design decisions and provide feedback to the developer about how the system should work. A prototype does not need to be fully functional; in fact it need only be partially functional to convey the intended meaning and allow a user to make an informed design decision.

In some cases, multiple alternatives to a proposed piece of functionality may be prototyped so that the user can choose which is most suitable to the desired outcome. Spending a few hours on a prototype in the early stages of a project may save many times that amount of time in rework later in the project.

Due to the flexibility and rapid development capabilities of TM1, prototypes can be quickly and easily developed. In many cases the prototype can be used as the first iteration of the system rather than just a throwaway decision making tool.

Through prototypes users get firsthand experience with the proposed system rather than a theoretical discussion of what might be possible. Prototypes are a key tool when developing systems using the more agile approach.

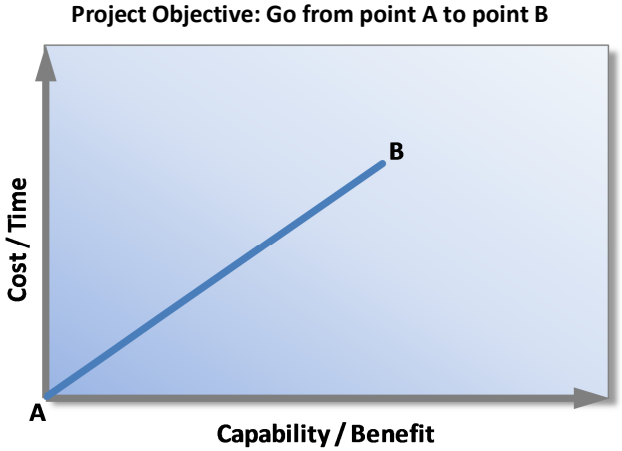
Achieving the Ideal Outcome

To further compare the more agile approach to other approaches, this section presents further discussion on how to achieve the ideal outcome when developing IT systems.

Of course, the goal of any project is to deliver a system that most closely fulfils the desired outcome. However, it is often the case that the ideal outcome is identified mid-way through the project rather than at the very start.

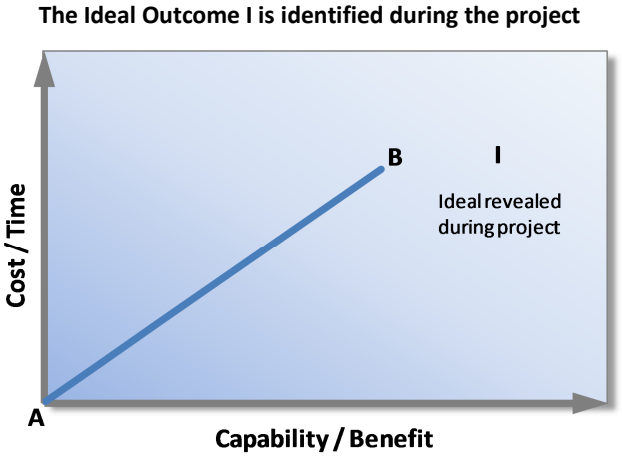
Using the Waterfall Approach

Using the waterfall approach, the system requirements and design are decided up front before development commences. This can be shown as follows:



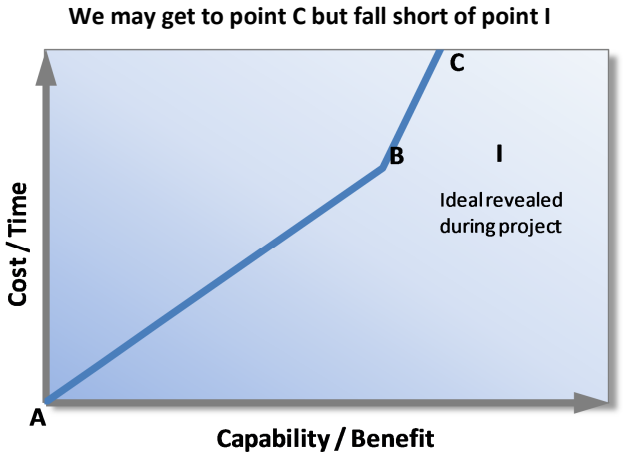
The vertical axis shows the cost/time required to go from point A to point B and the horizontal axis shows the capability of the system. As cost and time increase the capability will increase in a fairly linear fashion.

But what if the ideal outcome is identified midway through the project?



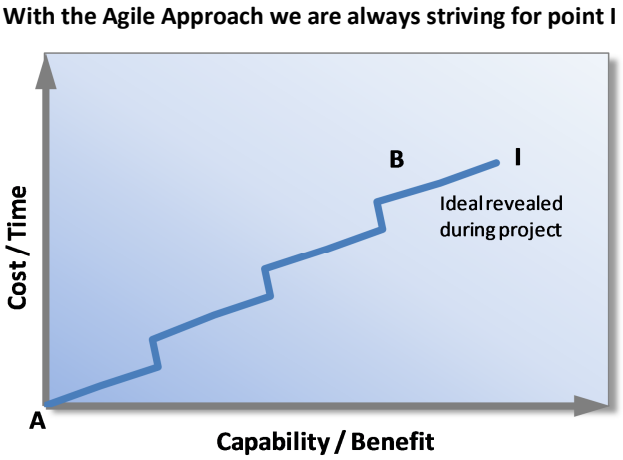
It will not be possible to achieve the ideal outcome I for the original cost and timeframe of the project due to the nature of the waterfall process. The changes will need to be managed by the change management process.

Furthermore, depending on how late in the project the ideal outcome is identified, it may not be possible to reach point I. It may be possible to reach some level of additional capability that falls short of the ideal outcome but this will likely come at a greater rate of cost and time due to the rework required as follows:



Using the More Agile Approach

With the more agile approach we are always striving for the ideal outcome. This can be demonstrated as follows:



Note that steady progress was being made during the first iteration. At the end of the first iteration some minor rework was required to ensure the system was heading towards the ideal outcome. This is shown by the capability of the system slipping backwards temporarily as the changes are being made. However the implementation is now on track for the ideal outcome and the adjustment has been made early in the project where the impact to time and cost are lower than if they were made towards the end of the project. Each subsequent iteration follows a similar pattern.

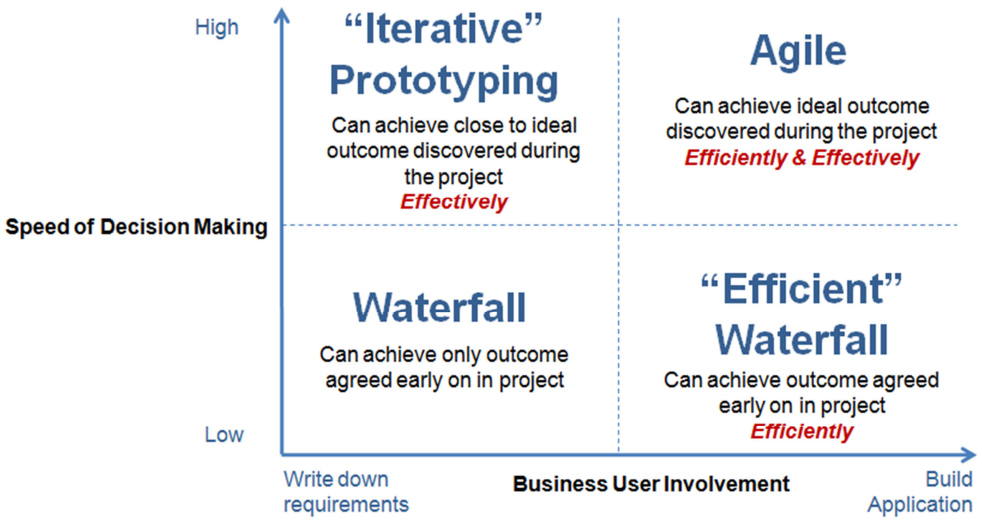
How "Agile Ready" is Your Project Team

The previous sections described the more agile approach and compared it to the waterfall approach to project management. However, there are certain characteristics that the project team members require to successfully deliver projects using the more agile approach.

As described earlier in this document, the more agile approach requires a collaborative approach to system development between end user and developer. This approach is most effective when the end users who are directly engaged with the project are:

- Empowered with decision making capability with regard to the project
- Have a very good understanding of the business problems that the system is to address
- Are heavily involved with the project throughout the implementation

If the project team is limited in one or more of these areas, then the effectiveness of this approach will not be as great. However, it's still possible to achieve some degree of agility even without all of these characteristics. See the following diagram for more information:



The vertical axis represents the speed of decision making that the users are capable of. This is a combination of how well they understand the underlying business problem to be solved and the degree to which they are empowered to make decisions on behalf of the project. The horizontal access represents their level of involvement in the project.

If the user representative has low levels of decision making capability and limited involvement in the project, then the waterfall approach will likely be the most effective approach. This approach will achieve the outcome agreed at the start of the project.

If the user has low levels of decision making capability but has a high level of involvement in the project, then the most practical approach is the "efficient" waterfall. Using this approach, the project will deliver the outcome agreed at the start of the project; however it will do it more efficiently as the user has greater involvement and interaction with the development team.

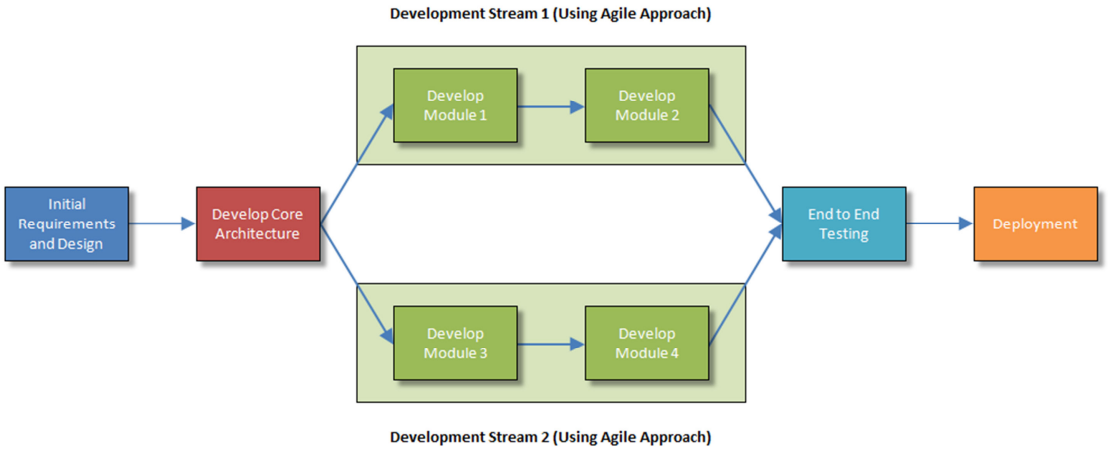
If the user has high levels of decision making capability but has a low level if involvement in the project, then the most practical approach is the "Iterative" Prototyping approach. This approach is similar to the more agile approach; however each iteration will be longer and developed with less feedback from the user. The reduction in the feedback loop will result in the developer needing to make more assumptions between iterations which may lead to a greater degree of rework in each iteration.

The ideal situation is where the end user has high levels of decision making capability and high levels of involvement with the developers. This arrangement will result in the most agile project approach and the greatest chance of achieving the ideal outcome.

Managing Large TM1 Projects

Additional consideration needs to be given to the project management approach for large TM1 projects. Large TM1 projects will likely see multiple development teams working in parallel on independent streams of work. Each stream would then be responsible for delivering specific "modules" of functionality.

In the case of a large project, the more agile approach is modified slightly as follows:



The project still commences with an initial requirements phase, however there is a new phase introduced immediately after requirements and prior or the iterative development phase commencing. The new phase is required to ensure that all development streams are working with a shared approach and understanding of the components of the system that will be shared, i.e. the core architecture. Core architecture components will likely include the following:

- Shared dimensions such as Time, Version etc
- Shared cubes such as system control cubes, FX cubes etc
- Security
- Naming conventions
- Presentation standards for reports, web sheets etc

Once all of the core architecture components have been established the iterative development process can begin independently in each stream. To ensure the maximum effectiveness of the more agile approach, each development team should be communicating regularly to ensure a consistent approach to the development of the system.

Once the iterative development phases have been completed, an additional end to end testing phase should be introduced. This ensures that all of the modules and core architecture components work together as a fully functioning system.

Keys to a Successful Project Using the More Agile Approach

- A flexible approach to project management does not mean an undisciplined approach. It is still important to manage each project phase to time and budget.
- Implement system using a "Best Practice" approach to development. Using best practice ensures the least amount of re-work when changes are required and leads to a robust and maintainable system.
- Use prototyping to help clarify and define requirements.
- Ensure key stakeholders are engaged throughout the project and make sure there is a strong feedback loop in place.