Planning an Iterative Development Project

What is Iterative Development

The traditional way of developing software consists of dividing projects into major phases such as:

1. Project Definition
2. Requirements Gathering
3. Analysis & Design
4. Development & Unit Testing
5. Integration, System Testing & Roll-Out

This approach is known as the "waterfall lifecycle". When applied to system development, it leads to two major drawbacks.

- **Late risk reduction**. Most of the risks remain alive until the system is assembled and tested at the end. There is a likelihood of rework being needed during the release phase to correct errors made during the early phases.

- **Resistance to change**. Requirements have to be understood up-front and not changed again. Rigorous change management procedures and systems are applied to "manage" (ie. Resist) change.

What if we could actually test the components of the system that present the highest risk before committing to a final design?
What would it be like if it we could overlap Analysis & Design with a significant portion of Development, Unit Testing and Integration?

Iterative development is an approach to the system development lifecycle that addresses these issues effectively. It can also be called "successive approximations".

Iterative Development does not only benefit software development projects. It can be a very effective approach for any project where the final outcome is not clearly defined up-front, and needs to be explored for, evaluating each attempt and re-directing the project as it progresses towards its final goal.

In iterative development the project is conducted not as a single "waterfall", but as a succession of "weirs", each of which tackles one part of the overall problem. These are repeated cumulatively until the entire problem has been resolved.

Each "Weir" (or iteration) consists of a complete end-to-end waterfall lifecycle. Each iteration has a defined scope which is validated at the end of the iteration. Testing in one iteration will take place prior to Analysis & Design in a later iteration.

Successive approximations in scope and objectives

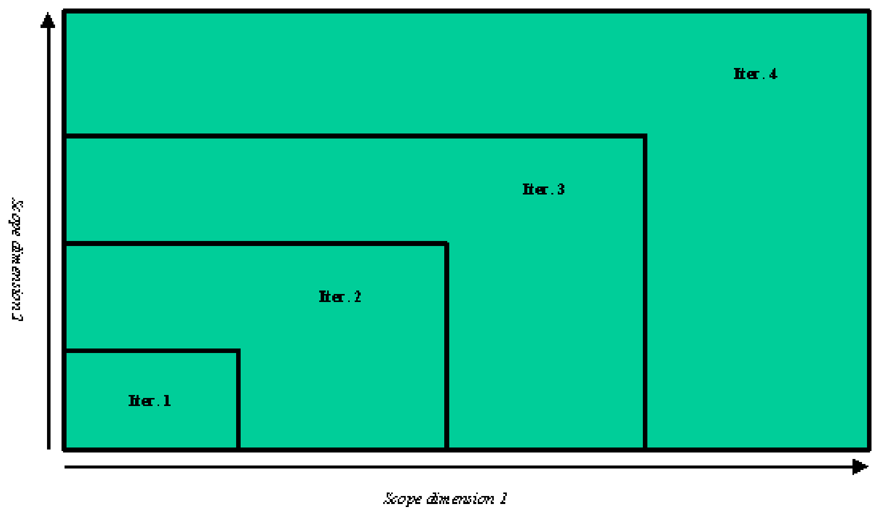

Figure 1

Advantages of the iterative approach include:

- **Early risk reduction:**

  Validation is included in each iteration, the most important components of the solution can be validated early before they are used for the rest of the project.

  Early risk reduction can be accomplished by using risk as one of the dimensions of scope, thus allowing to address first the components that carry the greatest risk.

- **Flexibility**:

  Each iteration includes Requirements Gathering and Analysis & Design. This allows to adopt an exploratory approach for those aspects of the solution that are not clear at the beginning.

  Flexibility can be accomplished by using functional decomposition (features, functions or use cases) as another dimension of scope.

Each iteration consists of a complete waterfall lifecycle. In early iterations there is greater scope for Requirements Gathering than for Integration and Roll-Out, whereas by the last few iterations all requirements will have been gathered, and most work will consist of Integration & Roll-Out.

Nevertheless, there should be an element of every discipline in each and every iteration.

As an example of an iterative process applied to software development, the **Rational Unified Process (RUP)** divides a project into four Phases and six Core Disciplines. The six core disciplines are:

1. Business Modelling,
2. Requirements,
3. Analysis & Design,
4. Implementation,
5. Testing,
6. Deployment

Each phase consists of one or more iterations focusing (not exclusively) on the following:

Inception Phase:	Project Definition, Business Modelling, Requirements
Elaboration Phase:	Requirements, Analysis & Design, Implementation
Construction Phase:	Implementation, Testing
Transition Phase:	Testing, Deployment

Other Disciplines are active throughout the project lifecycle, these are Project Management, Environment, Configuration & Change Management.

Each phase aims to achieve a set of milestones in terms of how much progress will have been made in each of the various disciplines.

The following diagram, taken from the Rational Unified Process documentation, exemplifies how each discipline performs a different scope of work at each iteration of the project.

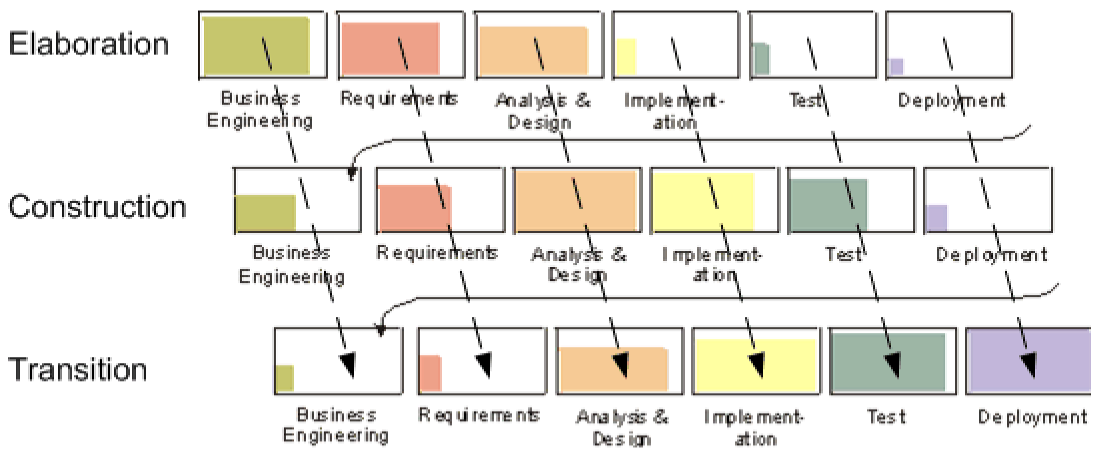

Figure 2

Breaking down a project into iterations.

The Project Manager starts planning an iterative project during project start-up. Prior to this a preliminary analysis or exploration might have been carried out. The project's overall scope and objectives are agreed, critical success factors and risks are identified and a candidate solution chosen. This information is outlined in the Project Brief, and detailed further in the Project Initiation Document (PID).

While developing the project plan it is advisable that the Project Manager work in close cooperation with the relevant Project Technical Manager, Business Project Owner and experts. It is also advisable that the Project Manager be familiar with the concepts and practice of iterative development, or obtain appropriate mentoring and coaching from the ITS Project Office.

The first step in developing the project plan is to determine the number of iterations and define the scope of each iteration. The minimum number of iterations is four, one for each of the phases described above, but it may be necessary to iterate one or more of the phases more than once.

Initial information gathering

In order to break a project down into different iterations, the following information is required.

1. The functional scope of the target system defined in such a way that it can be broken down into clearly identified components (business processes, major business functions, key business needs and features).

   If this information is not available then it is necessary to further define the project and perform some preliminary business modelling until this has been accomplished.

2. Non-functional requirements.

   Requirements such as performance, availability, reliability, usability, maintainability, capacity and physical location of users can serve to identify major technical risks of the project.

3. The Project Risk List.

   Must include business risks, project risks and technical risks.

   Some risks may be linked to the functional and non-functional requirements, for example some functional requirements may need more exploration or a technical requirement may generate a technical risk.

   If the risks are not sufficiently understood then it is necessary to conduct a risk assessment workshop with the principal stakeholders (including Project Technical Manager and Business Project Owner) in order to compile as complete a risk list as possible.

Choose the Iteration Pattern.

The Iteration Pattern describes how many iterations there will be for each phase of the project. This decision will depend on a number of factors, such as the overall size of the project and the degree of novelty in the project.

Note, the Iteration Pattern may have been already chosen as part of the Project Approach in the Project Brief

Choose the Prototyping Strategy.

The protoyping strategy is a definition of how many prototypes will be produced (if any), the scope of each prototype, which iterations they will be developed in, how they will be validated and what will happen to them after they have been validated.

Note, the Prototyping Strategy may have been already chosen as part of the Project Approach in the Project Brief

Determine the scope of each Iteration.

The scope of each iteration is defined in terms of the following:

- Risks (from the risk list)
- Major business functions, solution components or Use Cases.
- Scope of prototype (if any)

As a general rule one should:

- Assign the highest risks to the early iterations, so that they can be eliminated early
- Assign to early iterations the components or use cases that are directly linked to the selected risks or where the requirements are most likely to change
- Define the prototype scope so that it is aligned with the risks
- Use the prototype to also verify and establish critical components of the architecture

Iteration scoping needs to be performed in interaction with the main stakeholders, including the Project Technical Manager and the Business Project Owner, so that there is agreement.

Estimate overall project effort and phase schedule.

Although each Iteration can only be estimated reliably once it has been planned in detail, it is often necessary to provide a rough estimate of the total effort for the project and the schedule of the major phases and milestones prior to or during project start-up. At this point in time a detailed Iteration Plan is not available for every Iteration.

An approach available is to develop a first estimate and schedule based on the distribution of effort and time between phases which is typical for the chosen iteration pattern, the total estimate already provided in the proposal and more detailed estimates that can be developed for the early Iterations based on the information available.

Phases can be planned more accurately by taking into account the:

- Number of features, use cases or functions identified.
- Complexity of the features, use cases or functions already studied.
- Risks identified, both technical and business.
- Function-point, or use-case metrics.
- Result of any prototyping.

The following points need to be taken into account:

- If Iterations are time-boxed, there should be flexibility in terms of scope.
- Major milestones should correspond to phase transitions, not single iterations.
- The point of iterative development is to correct errors early, the earlier they are corrected the lower the impact in work and cost. Include a certain amount of re-work in your estimates for each and every Iteration, based on the size and complexity of the preceding Iteration.
- Round your resource estimates so that you can have critical resources and teams allocated on a continuous basis, rather start-stop involvement.
- Review your estimates with all critical stakeholders, work together with the them to ensure that there is agreement and commitment.

Planning an Iteration

Each Iteration is planned as if it was a project in its own right. A good approach to iteration planning is the product based approach prescribed by PRINCE2. This approach is synthesised as:

- Define the products that will be produced, and how they are interrelated (Product Breakdown Structure)
- Define the sequence in which the products can be produced (the Product Flow Diagram)
- Identify the activities and tasks required to create or transform each product
- Create a network of activities that reflects the Product Flow Diagram
- Identify resources required for each activity
- Estimate effort required for each activity
- Assign resources
- Develop schedule
- Analyse and improve schedule

Determine scope for each Discipline in the Iteration.

When planning an Iteration one must take into account that each discipline will have a different scope with respect to the other disciplines in the same Iteration (refer to Figure 2)

Examples

- The first (and only) Iteration of Inception may aim to complete 60% of Business Modelling, 40% of Requirements, 10% of Analysis & Design, 5% of Implementation, 3% of Testing (of prototype) and no Deployment.

- The last Iteration of Transition may aim to complete no Business Modelling, 1% of Requirements, 3% of Analysis & Design, 5% of Implementation, 15% of Testing and 70% of Deployment.

It is therefore necessary to develop a more detailed understanding of the scope that applies to each Discipline.

In order to do this, prepare a spreadsheet that has a row for every discipline (including Project Management, Environment and Change & Configuration Management) and also a row for every prototype that will be produced (if any) during the iteration.

In the first column, enter a brief and clear description of the use cases that will be covered by each core discipline. The description should also say something about the degree to which the use cases will be developed, for example “50% identified, 30% defined, 20% completed, 10% reviewed”.

If there are specific use cases that need to be realised in order to address a risk, then these should be referred to by name. Construction Iterations should always refer to specific lists of use cases rather than a generic percentage.

If there are one or more prototypes, enter in this column a description of the functional and/or technical scope of the prototype, for example “connectivity across the network” or “basic order entry”.

In the second column describe how each discipline and prototype will contribute to the mitigation of the risks in scope for the Iteration. This will depend on the specific risk mitigation strategies identified in the risk assessment workshop and documented in the risk list. These contributions should be described in terms of a specific outcome or “product”, for example “access to the system via the Internet” or “contract negotiated with software vendor”.

In the third column describe any other specific products that each discipline will contribute to the project during this Iteration. For example “development environment” or “project guidelines”.

Identify and classify the products of the Iteration.

It is strongly advised to carry out this activity in conjunction with the key project team members.

The products of each discipline, including Project Management, must be listed and classified,

Develop the Product Breakdown Structure.

The Product Breakdown Structure can be produced based on the list of products (including quality review products) and detailed further.

If a product has to go through a number of versions or iterations, each of these should be considered a separate product within the Product Breakdown Structure. For example, if you plan to produce three versions of a prototype, each of these is a separate product that will require specific activities to be performed.

The level of detail of the Product Breakdown Structure will depend on the level of granularity of the Work Packages used to manage product delivery and on agreement with the discipline Team Leaders.

A word of caution is due in relation to overlapping products between Iterations. The point of Iterative Development is to continually assess quality and to allow for a certain amount of re-work on an ongoing basis, correcting errors early. This means that the Product Breakdown Structure should include a significant number of review artifacts, and these in turn may create new revisions of previously existing artifacts, which should be considered as additional products.

Identify tasks and dependencies

An effective method to identify tasks and dependencies is the one It consists of first first producing a Product Flow Diagram, indicating the dependencies between the different products in the product breakdown structure and the sequence in which they need to be produced, and then assigning tasks to each product at the lowest level of the PBS.

This method is fully supported by the functionality of P2MSP, which is recommended to be used for this purpose.

Estimate work and resource requirements

For each task in the plan the effort required should be estimated, based on the type of resource required and the availability of the resources.

Assign resources, develop and optimise schedule and resolve conflicts.

By assigning resources to each role, each activity is placed on a time line. A schedule of work for the Iteration is developed, reviewed and optimised.

Review, publish and action the plan.

It is important to obtain commitment to the Iteration Plan by developing it in conjunction with relevant stakeholders, such as the project team members, sullpiers and participating stakeholders.

The plan should also be reviewed by the relevant Project Support (Project Review) roles, such as the ITS Project Office.

Additional points

Overlapping Iterations vs Phase Transition.

It is important to distinguish between overlapping Iterations and Phase Transitions.

Phases also correspond to the PRINCE2 definition of "Dividing a Project into Manageable and Controllable Stages". Stage transitions are management decision points, at which the Project Board (as defined in PRINCE2) reviews progress and commits resources and authority to spend for the next Stage. In this context, Stages (Phases in RUP) should not overlap, and there should be a clear management of Phase Transition.

Within a Phase, if there are multiple Iterations these can overlap. There are two reasons for overlapping Iterations:

- Providing continuous utilisation of resources from different disciplines.
- Allowing to start work early if it does not depend on work being carried out in previous Iteration.

The interdependency between Iterations needs to be well understood when overlapping Iterations. When overlapping Iterations it is advisable to perform an analysis on the dependencies between the product breakdown structures of the overlapping Iterations, to make sure that you are not planning to produce one artifact before the one that it depends on.