



**ITERATIVE DEVELOPMENT STANDARDS
VERSION 1.0**

MARCH 11, 2009

Information and Technology Management Branch

❖ IM / IT Standards & Guidelines ❖

Table of Contents

1. INTRODUCTION	3
2. DOCUMENT PURPOSE.....	3
3. INTENDED AUDIENCE.....	3
4. ITERATIVE DEVELOPMENT PROCESS	3
4.1 NUMBER OF ITERATIONS IN A PROJECT	4
4.2 DELIVERABLES IN AN ITERATION	4
5. ITERATION DEVELOPMENT STANDARDS AND GUIDELINES	4
6. APPLICATION INTEGRATION	5
7. ON-BOARDING PROCESS FOR ITERATIVE APPROACH APPLICATIONS	5
8. SOME ISSUES WITH THE ITERATIVE DEVELOPMENT APPROACH.....	6
9. ITERATIVE DEVELOPMENT RECOMMENDED PATTERNS.....	7
10. ITERATIVE DEVELOPMENT ANTI-PATTERNS	9

1. Introduction

The basic idea behind *iterative approach* is to develop a software system iteratively and incrementally, allowing the developers and users to take advantage of lessons learnt during the development of earlier iterations of the system development. In *Iterative Development approach*, the whole process of system development typically iterates through all the phases of System Development Life cycle (SDLC), starting from gathering requirements to delivering functionality of a working release. Key steps in the iterative development process are to start with a simple implementation of a subset of the software requirements and iteratively enhance the evolving software versions until the full system is delivered/implemented. In each iteration (release), modifications to design/architecture and code are made as the new functionality is added.

2. Document Purpose

The purpose of this document is to define and describe standards for Iterative development approach for the Ministry applications. These standards are to be used in conjunction with the existing Ministry ADE standards.

3. Intended Audience

The main intended audiences for this document are the ITMB Team leads/Business Analysts, IMG staff and Application Development service providers who are involved in the application design/architecture, development and implementation process. The AMS service provider also would find this document helpful in familiarizing Ministry's iterative development standards.

4. Iterative Development Process

Iterative development approach slices the deliverable business functionality into iterations. In each iteration a slice of functionality is delivered through the SDLC, starting from the requirements through to testing and deployment. Each iteration thus is divided into the same SDLC phases : requirements, architecture/design, construction, test and deployment. Each iteration is time-boxed rather than feature-boxed. Architects and analysts usually work one iteration ahead of developers and testers to keep up with the pace of system development and release.

The major goal of Iterative Development approach is to provide the customer with some working model of the system at an early stage of the development cycle and to reduce the time to delivery of usable functionality. **As new functionality is added to the system, a new iteration (release) is launched which should have fewer bugs and more features than the previous release. Typically, the ability to derive business value from the**

deployment of any single release should be independent of the content of any subsequent release.

The following sections describe the Ministry ADE Standards on the Iterative Development approach.

4.1 Number of iterations in a project

Ministry ADE process recommends that High risk category projects should have a **maximum of three iterations** (releases) in general. The actual number of iterations however could vary from project to project depending on the business situations. For low/medium risk projects, it is at the discretion of the project teams and Clients to decide on the number iterations – one, two or three iterations.

4.2 Deliverables in an iteration

Ministry ADE process recommends that all the ADE SDLC deliverables of the chosen roadmap must be produced in each iteration. A single version (copy) of the deliverables will be maintained for a system and the deliverables will be modified in each subsequent iteration. There must be adequate documentation embedded inside the deliverable about the modifications / improvements done in the iteration over the previous iteration. For document template deliverables, the “Revision log” section should specify what improvements/changes are made to the document in the current iteration. For executable deliverables, the corresponding source code must contain comments for each iteration indicating the iteration development/implementation date, salient features of the iteration briefly describing what changes/enhancements to the previous business functionality were introduced during the iteration. The comments also should describe whether the changes/enhancements were implemented by way of changing application architecture, by way of changing database schema etc.

5. Iteration Development Standards and Guidelines

Ministry ADE process recommends the following standards for Iterative development approach :

1. High risk category projects should have a **maximum of three iterations** (releases) as a general guideline. The actual number of iterations however could vary from project to project depending on the business situations. For low/medium risk projects, it is at the discretion of the project teams and Clients to decide on the number iterations– one, two or three iterations.
2. Duration of each iteration is typically between 6 to 12 weeks and must not exceed 3 months. For projects that are over 1 year duration, the number of iterations

- could be increased with prior discussions and approval from Ministry Architecture Committee (MAC).
3. Each iteration must be time-boxed, and not feature boxed. Example : For an iteration of 12-weeks duration, define the business functionality that can be delivered in 12-weeks time including the associated SDLC deliverables.
 4. Document and signoff the High-level Business requirements upfront before the development begins.
 5. Build functional prototype for system early in the development.
 6. Divide the detailed business requirements, design, test and implementation phases into multiple iterations. Each iteration must encompass all the phases of the SDLC : detailed requirements, design, build, test and transition into production.
 7. If there is a need for user training in a specific iteration(s), it should be addressed and provided for in the project.
 8. Baseline an executable architecture early in SDLC.
 9. Develop one or more business functionality/subsystems at a time.
 10. Develop the most critical scenarios (use cases) first.
 11. Choose those use cases for development that must be delivered during the timing of the iteration.
 12. It is a common practice that Architects and Analysts work one iteration ahead of developers and testers to keep up with the pace of system development and release.
 13. A single version (copy) of the deliverables must be maintained for a system and the deliverables will be modified in each subsequent iteration.
 14. A single copy of Project Management deliverables (such as Project work plan, MPP etc) must be maintained. Within the project work plan, the work plan for individual iterations (releases) can be elaborated.
 15. There must be adequate documentation embedded inside the deliverables about the modifications / improvements done in the iteration over the previous iteration.

6. Application Integration

If the application being developed is to be integrated with other application(s), then the decision to integrate by iterations or the whole completed application will be taken by the project team depending on the situation.

7. On-boarding process for Iterative Approach Applications

Individual iterations will not be on-boarded due to cost and time factors associated with the on-boarding of each iteration. After all iterations are released into production, the whole application will be on-boarded to the AMS provider.

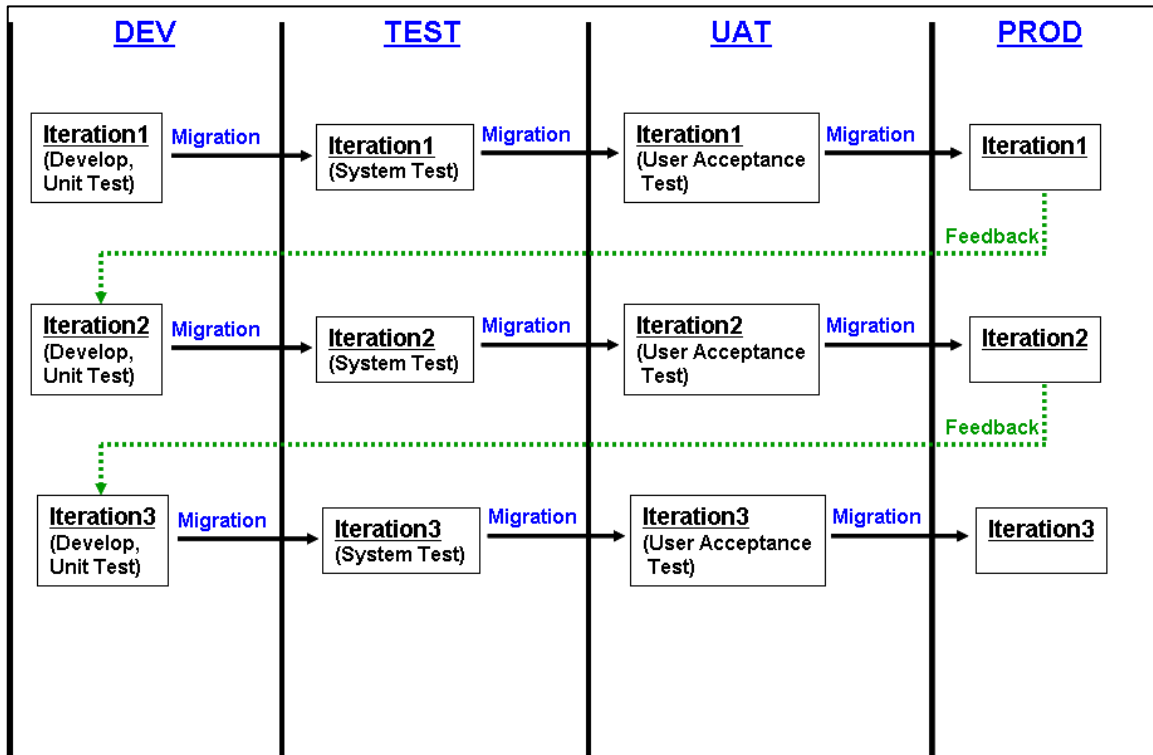
8. Some issues with the Iterative Development approach

1. QCIL review of deliverables needs to be done for each iteration of the project. This is likely to indicate that there will be more effort and time for QCIL reviews compared to the waterfall approach deliverables. However, iterative approach deliverables are expected to be light-weight and thus QCIL review is likely to be much less onerous than the waterfall approach deliverables.

9. Iterative Development recommended patterns

Typically, Iterative Development Approach should follow the following patterns :

Pattern 1



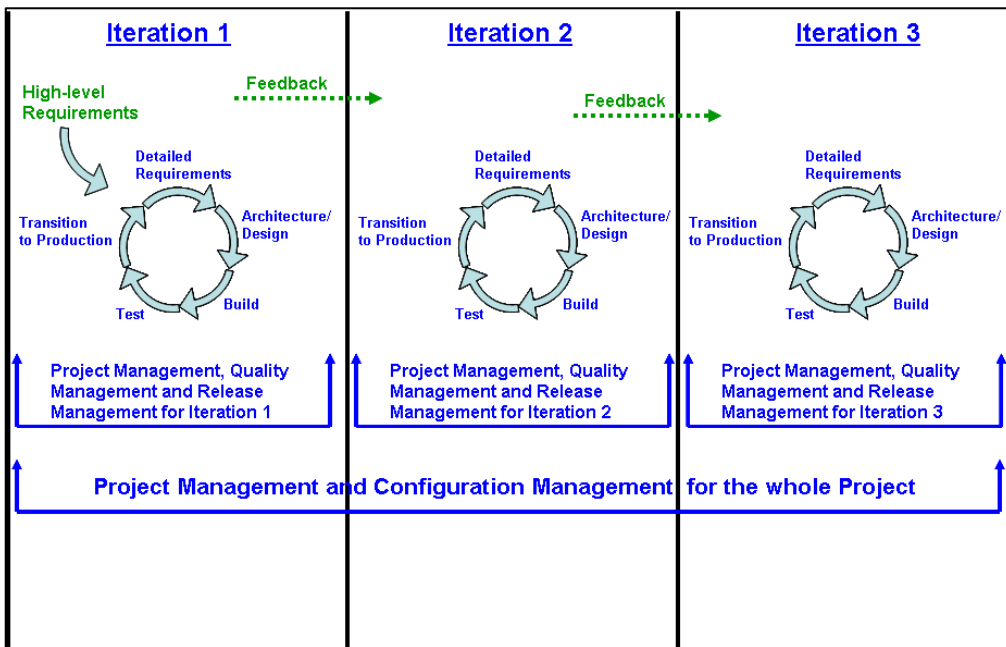
Pros

1. Application is delivered early, delivered often through each iteration.
2. Each iteration delivers production version of the application.
3. Early iterations expose risks and this helps in risk mitigation before subsequent iteration starts.
4. Opportunities for reuse are very high, with earlier iterations focusing on development of reusable code/functionality that could be reused in subsequent iterations.
5. Testing occurs early and often resulting in manageability of testing tasks and test plans.
6. Enables continuous communication and engagement with clients and stakeholders throughout the application development.
7. Work load on QCIL review, code walkthrough and UAT testing are much less due to individual iterations being of manageable size.

Cons

1. Maintenance phase of the Application starts right after the first iteration is in production. This calls for some kind of Service Levels in place by the time the first iteration is in production.
2. Resource availability for the development, testing and production transition must be well-defined due to iterations getting ready for production one after another contiguously.
3. QCIL review of deliverables is to be done for each iteration as opposed to one time review for the whole application.

Pattern 2



Pattern 3

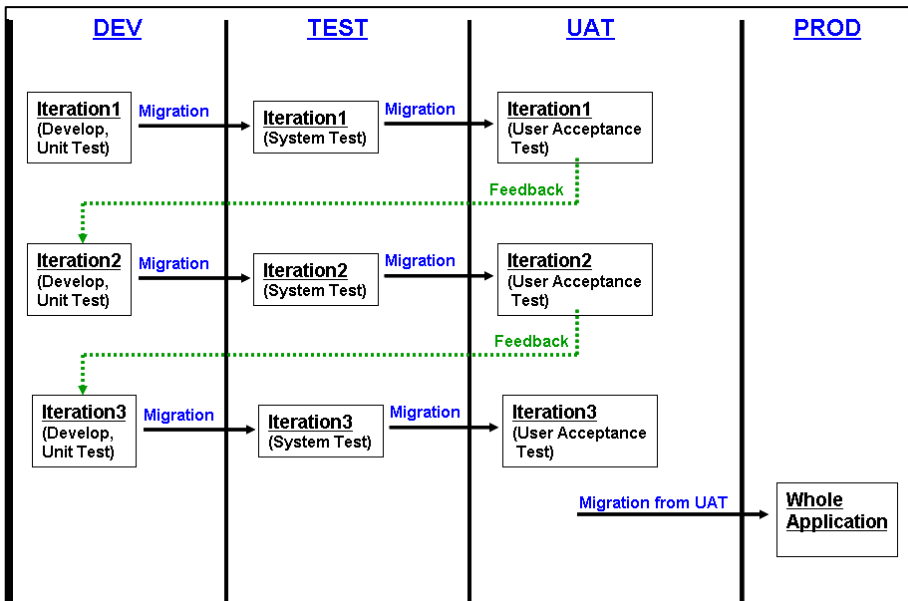
Task	Stage
Project Planning	Application/Project
Iteration Planning	Each iteration
High-level Business Requirements	Application/Project
High-level Technical Requirements (if applicable)	Application/Project
Detailed Business Requirements	Each iteration
Baseline Architecture (if applicable)	Early in the project – before the Iteration 1 starts
Technical Architecture/Design	Application/Project
Application Architecture/Design	Each iteration
Development/coding	Each iteration

Unit Testing	Each iteration
System Testing	Each iteration
UAT Testing	Each iteration
User Training	Depends on Specific iteration or Application
QCIL review of deliverables	Each iteration
Release Management/Transition into Production	Each iteration
Review and Feedback	Each iteration

10. Iterative Development anti-patterns

Projects desirous of adopting Iterative Development Approach should avoid the following anti-patterns :

Anti-Pattern 1



Pros

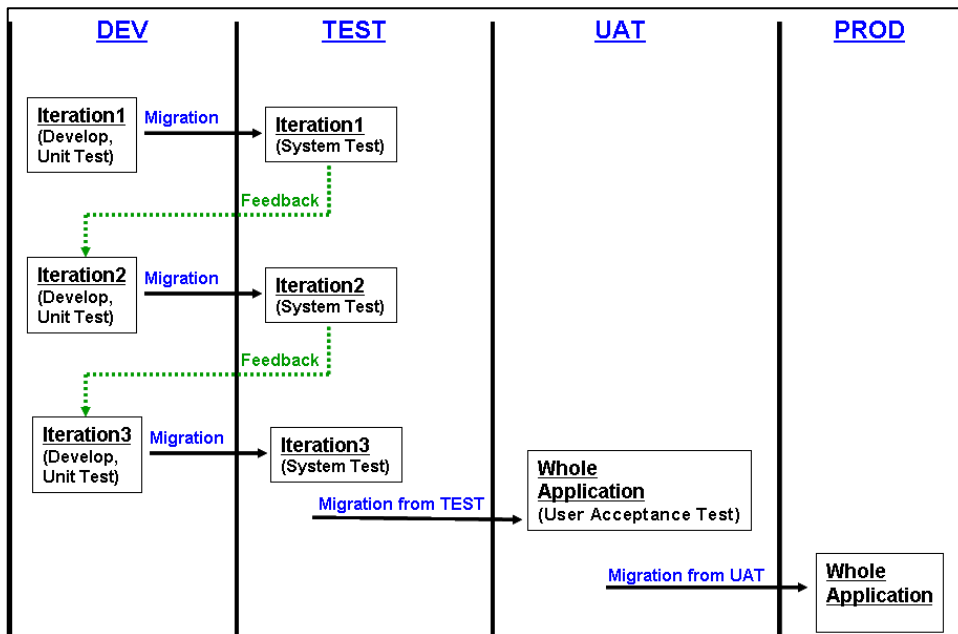
1. The Maintenance phase starts just after the Production rollout of the whole application from UAT and thus Maintenance teams are not bugged by iteration-level maintenance issues.

Cons

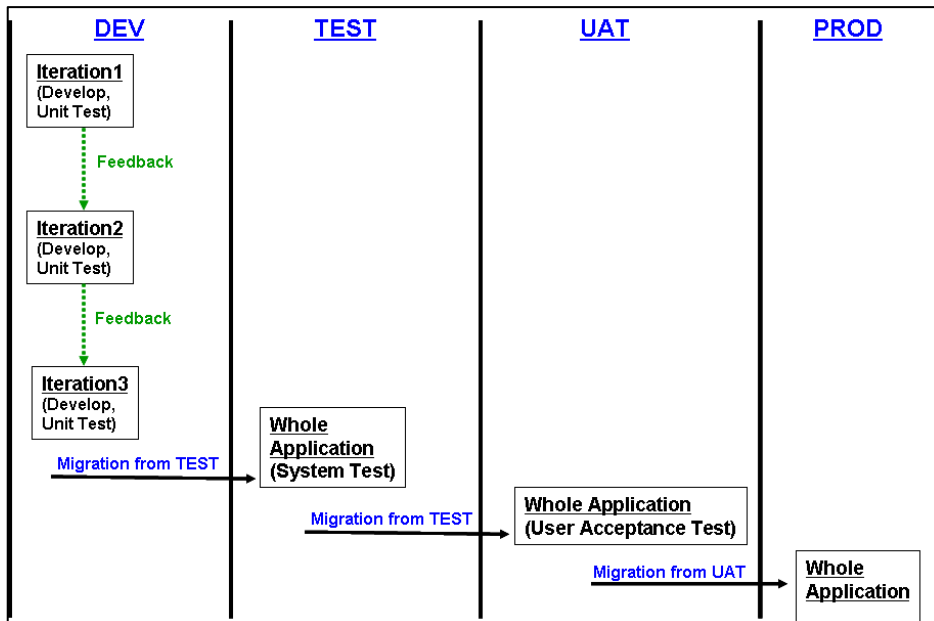
1. Clients (particularly the “public”, for public facing/internet application) will not have opportunity of seeing and using some critical functionality of the system early and they have to wait until whole application (all iterations together) is rolled into production, which is contrary to the principles of Iterative approach.

2. If the application fails in Production, it is difficult to trace which iteration caused the failure and fixing the bugs also becomes time consuming.
3. No opportunity for tracking and implementing “Lessons learnt” early in the project in a progressive manner from previous iterations.
4. UAT testing of iteration(s) alone is not sufficient to get realistic feedback on iteration’s business value and user experience. The iteration needs to be moved into production and needs to be used in real-life situations by real users/clients/public to get realistic feedback on each iteration from the Production users.

Anti-Pattern 2



Anti-Pattern 3



Anti-Pattern 4

