

COUNTY OF FRANKLIN, VIRGINIA

# Systems Development Methodology

---



Version 1.0

40 E. Court Street • Rocky Mt., Virginia 24065  
Phone 540.483.3030 • Fax 540.483.1045

# Table of Contents

Phases of Development .....	2
Definition Phase .....	2
Objectives .....	2
Artifacts.....	2
<hr/>	
Analysis & Design Phase.....	3
Objectives .....	3
Artifacts.....	3
<hr/>	
Construction Phase .....	3
Objectives .....	3
Artifacts.....	4
<hr/>	
Implementation Phase.....	4
Objectives .....	4
Artifacts.....	4
<hr/>	
Post-Implementation Follow-up .....	4
Objectives .....	4
Artifacts.....	5
<hr/>	
Roles and Responsibilities .....	6
Analyst Role.....	6
Developer Role.....	6
Tester Role .....	7
Manager Role .....	7
Stakeholder (End Users).....	8
Disciplines .....	10
Business Modeling.....	10
Requirements.....	11
Analysis & Design.....	12
Test .....	12
Deployment .....	13
Configuration & Change Management .....	14
Project Management .....	15
Artifacts .....	17
Requirements Artifacts .....	17
Project Definition Document.....	17
Glossary.....	18
Prototype.....	18
Software Requirements Features List.....	18
<hr/>	
Analysis & Design Artifacts .....	19
Data Model .....	19
Business Rules Document .....	19
<hr/>	

Object Model .....	20
<hr/>	
Test Artifact Set .....	20
Test Plan .....	20
Test Case .....	21
Test Case Validation Log.....	21
<hr/>	
Implementation Artifact Set.....	21
Deployment Plan .....	21
End User Support Materials .....	22
<hr/>	
Configuration and Change Management Artifact Set.....	22
Change Management Plan .....	22
Change Request .....	22
Change Issue Log.....	22
<hr/>	
Project Management Artifacts .....	22
Iteration Plan.....	22
Risk List.....	23
Product Acceptance Plan .....	25
Project Acceptance Report .....	25
<hr/>	
Appendix A – Glossary .....	26

## Revision History

<b>Date</b>	<b>Version</b>	<b>Description</b>	<b>Author</b>
3/17/03	1.0	Initial draft	Sandie Terry
4/17/03	1.1	Modified to clarify the end user roles and add items to the glossary. Based on feedback from I/T staff.	Sandie Terry
5/26/04	1.2	Modified to include the development and use of an object model beginning with the Analysis & Design phase through Implementation	Sandie Terry

# Phases of Development

*The steps to be taken to achieve a feasible design and a solid system to implement.*

Systems development flows through several different phases with each phase providing the basis, or details, to proceed to the next phase. This section will briefly outline the phases to be contained in each systems development project and the objective of each of those phases. Later sections in this document will provide the details, deliverables and responsibilities of each role for these phases.

## Definition Phase

### Objectives

Primary objective is to attain concurrence from all project stakeholders on the objectives of the project. This is particularly important for new development or re-design/rewrite of existing systems. Even for projects that will deliver enhancements to existing systems, this phase will ensure that everyone agrees on the project objectives and the acceptance criteria. The primary objectives of the Definition Phase include:

- Establishing the project scope – what will be included and what specifically will be excluded.
- Defining the acceptance criteria – specifies the details that must be present to declare the project a success.
- Estimating and documenting potential risks.
- Estimating a schedule for the entire project which will be adjusted and detailed during later phases.
- Determine, and possibly demonstrate, candidate architecture for the solution.

### Artifacts

- The **Project Definition** document which will define the project's core requirements, key stakeholders, constraints, assumptions, product features (high-level) and change request process.
- The **initial iteration project plan** that will include time frames and resource requirements. The following phases will only be included in skeleton format to be detailed in later phases.
- **Risk list** which will outline any risks that may impact the project's success.
- **System Requirements Features List** which details the functional requirements.
- **Product Acceptance Plan** to detail acceptance criteria and responsibilities.
- **Iteration Plan** defines what will be designed/developed during the next iteration.

- **Project Glossary** which defines terms specific to the project / system and is updated throughout the life of the project when necessary.
- **Configuration Management Plan** describes the tools, format and process associated with managing the change control process throughout the life of the project.
- **Change Issue Log** used to capture all changes, issues or problems not tracked through the Change Control process that impact the management of the project.

## Analysis & Design Phase

### Objectives

Primary objective of this phase is to set the development/project architecture. If the architecture consists of new development components, then a prototype may be required to essentially serve as a proof of concept. The development and production environments are established.

This phase will also refine the Project Definition based on information discovered during this phase. The project plan will be expanded to include the detailed iteration plans for the Construction phase.

### Artifacts

- One or more executable **prototypes** to demonstrate a solid system architecture.
- **Data Model** – major data model elements (entities, relationships and tables) defined and reviewed.
- **Object Model** - describes the realization of use cases, and serves as an abstraction of the implementation model and its source code.
- **Iteration Project plan** – updated and expanded to include detailed iteration plans for construction.
- **Risk list** – updated and reviewed.
- **Business Rules document** – details and records all defined business rules.
- **Iteration Test Plan** that will outline the primary test targets and dependencies.
- **Scope Change Log** to track and manage scope changes.

## Construction Phase

### Objectives

This phase will finalize the design by clarifying the details and actually developing the system as defined during the previous phases. Development should be accomplished iteratively and incrementally as efficiently as possible. This requires balancing resources, work and quality to deliver the right solution as soon as possible.

#### Artifacts

- The “**system**” which means an executable application as described in previous deliverables.
- Updated **Test Iteration** and **Test Cases/Scripts** documents that outline the test scripts/plans.
- Updated **data model** to include additional elements required to support the developed system.
- **Object Model** is refined by detailed design decisions during the construction phase.
- **Deployment Plan** document detailing requirements to successfully deploy the developed system.
- **Test Validation Log** provides details of test dates and results as tests are performed by both the developers (unit and system tests) and the users (acceptance testing).
- **Scope Change Log** to track and manage scope changes.

## Implementation Phase

#### Objectives

The objectives of this phase are to primarily prepare the system for implementation – installation into the production environment. The User community must now be totally focused on final testing while the developers fine tune the system based on user feedback. The project is nearing the point of closure, even while another project for the same system (either enhancements or another phase) may be starting. If the new system is replacing an existing production system, then final testing may actually parallel the current system to ensure all data and reporting is correct. Any user training that is required is provided during this phase.

#### Artifacts

- The “**system**” in accordance with the project requirements and usable by the customer.
- **Training materials** which may include user documentation and/or training workshop materials.

## Post-Implementation Follow-up

#### Objectives

Once a system has been implemented into production, the entire system, a follow-up will be conducted with the primary end users to ensure that the system is performing as requested and functions properly. Any problems or issues must be addressed and resolved.

A by-product of every software development project is an enhancements list. This list will consist of system enhancements or modifications that were requested either during the project or just after implementation. These items will become the basis for a future enhancements project.

#### Artifacts

The details of the project and any outstanding requests will be included in the Project Acceptance Report. This report shall be delivered by the Project Manager to the Project Sponsor within two weeks of system implementation. If the newly developed system must parallel an existing system for some time period, then this report will be delivered within two weeks after the existing system has been turned off.

The customer (Project Sponsor) must sign off on this document which will serve as acknowledgement that Information Technology has completed this project.

# Roles and Responsibilities

*Roles are typically filled by an individual, or a group of individuals, working together as a team toward a common goal.*

**R**oles are not individuals, but rather the activities and responsibilities individuals must perform within the business environment. A project team member can actually fill more than one role, for instance, one member could take on the role of developer and technical writer. Every role in a project is responsible for specific activities and artifacts. Activities generally require and/or produce artifacts (documents, diagrams, charts, etc.).

## Analyst Role

The analyst is primarily involved in eliciting, investigating and documenting the business requirements. The following artifacts are the responsibility of the analyst role. However, the end users or customers are responsible for providing the information to formulate the Business Rules Document, the Software Requirements Features List and the user interface prototype.

- Business Rules Document
- Software Requirements Features List
- Project Definition Document
- Test Iteration Plan
- User Interface Prototype
- End-user documentation and input on Training Materials.

## Developer Role

The developer is primarily involved in designing and developing the software based on the documented requirements and business rules. The following artifacts are the responsibility of the developer role:

- Change Requests if the design dictates a change to the original features list.
- Data model.
- Object Model
- Development of the “system”.
- Test Cases/Scripts.

- Execution of test plans in the unit and system testing tasks.
- Prototype(s)
- Test Environment configuration – although not really a document, this is a major deliverable.

## Tester Role

This role is responsible for the primary activities required to completing testing of the newly developed or enhanced system. These activities include conducting the defined test scripts and logging the test results. Tests are performed during development as unit and system tests (by the developers) and then prior to implementation as part of user acceptance testing (by the user). The following artifacts are the responsibility of the tester role:

- Test Validation Log provides test dates and test results.
- Change Requests when system changes are identified during execution of the test scripts.

## Manager Role

The Manager role is mostly involved with configuring and managing the software development and implementation effort. Although one individual can take on multiple roles, it is most important to try to ensure that the individual taking on the role of Manager is not the same individual doing the system development. The following artifacts are the responsibility of the Manager role:

- Configuration Management Plan – defined late in the Project Definition phase to define the procedures for managing the change requests throughout the life of the project.
- Change Request – updated or deleted when confirm duplicate or rejected Change Request, or new ones discovered during Test Iteration Plan development.
- Project Definition Document – specifically the Change Request process and format.
- Change Request – responsible for scheduling reviews of submitted Change Requests for approval or rejection by the Change Control Board.
- Project Status Reporting – includes project plan updates, risk mitigation, project issues, etc.
- Deployment Plan – documenting how and when the product will be available to the user community.
- Manage Acceptance Testing – although not really a document, an integral part of the success of the project to ensure all testing is meeting defined acceptance criteria.

- Product Acceptance Plan – written procedure agreed by customer and development team for determining the acceptability of the product.
- Risk Lists – identify and assess risks.
- Iteration Test Plan – detailing test scripts to be executed by developers and users.
- Training Materials – created with input from Analysts and Developers.
- Iteration Plan – detailing exactly what will be designed/developed during each iteration.
- Change Issue Log – to track and manage changes or issues not managed through the Change Control process.
- Project Acceptance Report – serves as the official acknowledgement from the end users that the project has been completed as defined.

## Stakeholder (End Users)

This role is involved in support and other miscellaneous tasks of the project. Basically this role identifies anyone that is materially affected by the product. Although not listed, the end users (who are also considered Stakeholders) are responsible for providing information necessary to prepare many of the other project artifacts. In fact, the end users are heavily involved in all of the following phases:

- Definition Phase
- Analysis & Design
- Implementation
- Post Implementation

The artifacts this role is directly responsible for include:

- Change Request – whenever a necessary change is identified as a result of reviewing artifacts, prototypes or during testing.
- Change Request – review and resulting approval or rejection of submitted Change Requests.
- Test Validation Log – details of testing including dates and results.

The artifacts this role is indirectly responsible for include:

- Project Definition Document
- Software Requirements Features List

- Business Rules Document
- Product Acceptance Plan
- User Interface Prototype
- Project Acceptance Report

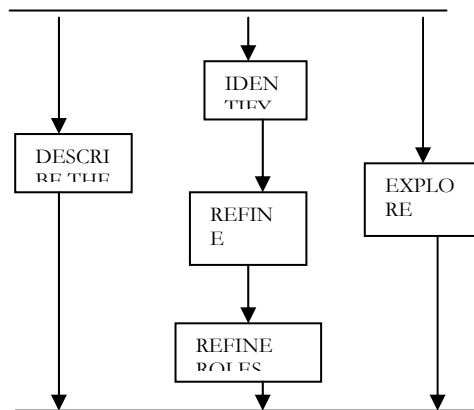
# Disciplines

*A discipline includes all activities a role or roles may go through to accomplish a particular piece of work which usually includes creating and/or using various artifacts.*

The following disciplines will provide the purpose of the particular activity (or activities) and how they relate, along with the workflow, artifacts involved and some guidelines that may be followed. The workflow provides the typical sequence of steps taken to accomplish a particular activity or group of activities.

## Business Modeling

The purpose of business modeling is to understand the structure and the dynamics of the organization for which a system is being developed and deployed. It also serves to understand current problems in the organization and ensure that customers, end users and developers have a common understanding of the organization. During the process, system requirements are defined to support the organization. Many of the following disciplines are driven from the information gathered during this initial process.

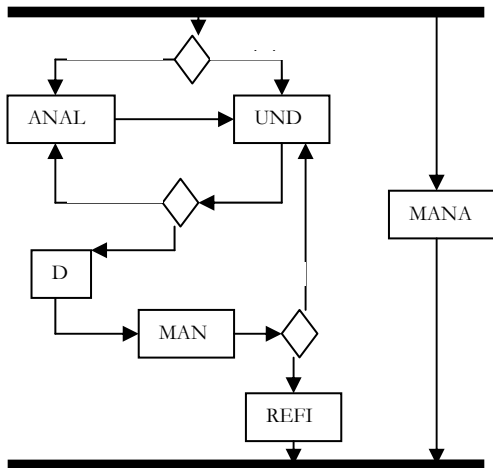


Business modeling generally results in Business Process Model diagrams and provides the beginnings for the Business Rules Document. The Glossary document may also be started at this time as the business partners define terms during the modeling sessions.

## Requirements

Requirements are defined to establish and maintain agreement between the customers and the developers as to what the developed system should do. Requirements define the *boundaries* of the system, what is specifically to be included in the functionality of the system and what is specifically excluded from the system. Requirements are used to determine the effort involved in designing and developing the system and are the basis for scheduling the project iterations. The requirements also provide the basis for designing a user-interface for the system, focusing on the needs and goals of the customer.

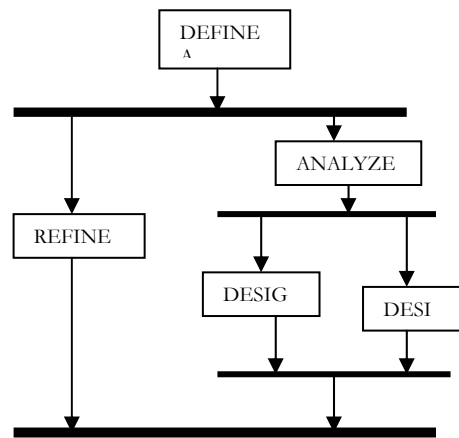
Stakeholders' requests are elicited and gathered during Requirements. This may occur during individual interviews or during a joint session. This provides a "wish list" of what all stakeholders want to be included in the developed system.



Requirements use the Business Process Models and the Business Rules document and generate the Project Definition Document, the Software Requirements Features List, and a user-interface prototype of the proposed system. Generally the Glossary is updated with new terms or started if one has not already been created. Requirement changes and change requests are usually beginning to be gathered and recorded in some fashion (possibly a spreadsheet or a database).

## Analysis & Design

Analysis and design transforms requirements into system features. The design adapts the features for the type of environment the system must run in and for performance. This discipline includes designing the database and the user-interface as well as any interfaces to other systems.

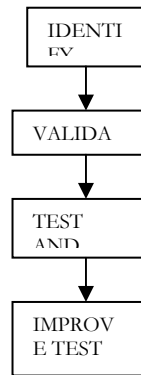


Requirements provide the primary input for this discipline which then produces a Data Model and possibly an architecture or analysis/design model.

## Test

Testing focuses primarily on the evaluation of quality by finding and documenting defects in the software, providing the validity of assumptions made during analysis and design, validating the software functions as required and validating the requirements have been implemented properly. All other disciplines are focused on improving the quality of the developed system, whereas

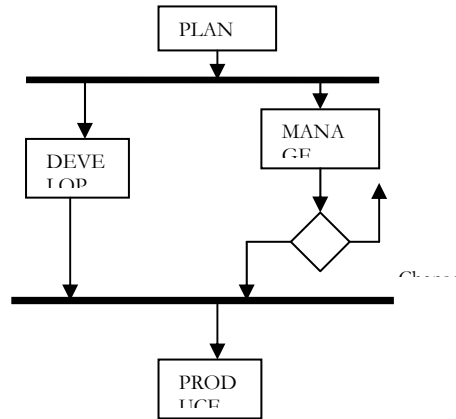
testing is focused on identifying the problems or weaknesses. Testing typically accounts for 30-50 percent of the total software development costs. The different ways a software program may behave are unquantifiable, therefore testing is extremely difficult. A continuous approach to quality, initiated early in the software lifecycle, such as following a well-defined methodology, may greatly reduce this cost while improving overall quality.



This discipline uses artifacts from most all of the previous ones, relying on the requirements, the features list and the design models. Test plans are followed to complete test cases and the test case validation log.

## Deployment

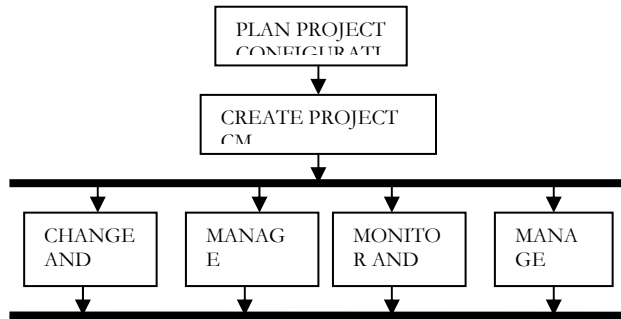
The Deployment discipline describes the activities required to ensure the developed product is available to the end users. Although the majority of the deployment activities occur during the Implementation phase, many of the artifacts are produced early in the project to plan and prepare for deployment.



The requirements are key to producing the End User Support material and the Test Plan and Test Cases are used during acceptance testing. The Deployment Plan is used to schedule activities and plan resources for the actual deployment of the finished system.

## Configuration & Change Management

Configuration and change management controls maintain the integrity of the project's artifacts. This involves identifying configuration items, restricting changes to those items, auditing the changes made and defining and managing configurations of those items.

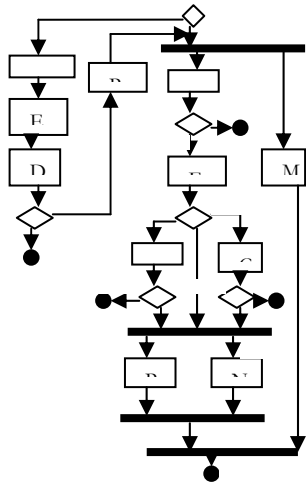


The Configuration and Change Management (CM) system is used throughout the project lifecycle to manage and control the project's artifacts. The artifacts that are directly related to this discipline include the Configuration Management Plan, the Change Request and the Change Issue Log.

## Project Management

Project Management is the art of balancing competing objectives, managing risk and overcoming constraints to deliver a software solution to the business that meets the customers' needs. Project management provides a framework for managing software intensive projects, provides guidelines for planning, staffing, executing and monitoring projects, and a framework for managing risk. All other disciplines are utilized as part of the project work.

Artifacts for this discipline include Iteration Plan, Project Plan, Risk List, Product Acceptance Plan, the Change Issue Log and the Project Acceptance Report.



# Artifacts

*Documents or models used to clarify communication between users of the system and designers of the system.*

Artifacts are either complete when first created or are modified during phases of the project, as a living document. Artifacts can be documents that detail requirements, features or test scripts or they can be models that depict a database, component integration or architecture. The artifacts are grouped into sets corresponding to the various disciplines involved in software system design and development, however, some are used in a number of disciplines.

## Requirements Artifacts

### Project Definition Document

Defines the stakeholders' view of the product to be developed and specified in terms of the stakeholders' needs and requirements. This document states the purpose and scope of the project including the problem statement and the business opportunity. The document also provides the list of features that the developed product must provide. A *feature* is defined to be a piece of functionality that is quantifiable in user terms, such as “*shall report all licenses sold during a specified period*”.

The Project Definition Document also specifies environment and documentation requirements. Change control procedures and the participants of the Change Control Board are also defined in this document.

Basically the PDD becomes the contractual agreement between the designers and the end users as to the product requirements.

To ensure a complete PDD, consider the following:

- Have you fully explored what the “problem behind the problem” is?
- Is the list of stakeholders complete and correct?
- Does everyone agree on the definition of the system boundaries?
- Have constraints to be put on the system been sufficiently explored – political, economical and environmental?
- Will all of the key features identified and defined solve the problem(s) identified in the problem statement?
- Are the features consistent with the constraints?

## Glossary

The glossary simply defines terms important to the project and is created to ensure everyone is speaking the same language. Often in business a term can be used to mean different things by different people, lending terminology to the root cause of communication misunderstandings. The glossary is used to prevent those misunderstandings and to ensure consistent use of terms.

To ensure a complete glossary, consider the following:

- Is every term clearly and concisely defined?
- Are terms used consistently throughout the various project artifacts?

## Prototype

The prototype is built primarily to detail and design user interface requirements. Usability is vital to a successful system design and it is critical to the successful implementation of any software system. The prototype can be a web prototype that identically imitates the developed product or may be as rudimentary as sketches on paper. The important point is the usability requirements have been discussed, defined and demonstrated in some format.

“A user interface is an interface that enables information to be passed between a human user and hardware or software components of a computer system.”  
[IEEE, Std 610.12-1990]

## Software Requirements Features List

This artifact captures the complete requirements for the software to be developed although it should not describe any design or implementation details. All requirements are broken down into client valued pieces of functionality – a feature. Each feature is described in detail including usability, reliability, performance and supportability requirements. Design constraints are identified and defined as discovered. Requirements are stated for online user documentation and the Help system.

All interfaces to existing or planned systems/components are identified and detailed as additional requirements of the product to be developed. These interfaces include user interfaces, hardware interfaces, software interfaces and communication interfaces.

The document should include licensing requirements and any applicable standards.

Consider the following when developing the SRFL:

- Are all requirements stated truly software requirements?
- Are the requirements unambiguous – having only one interpretation, stated in the end-user’s language, and have diagrams been used to augment complex descriptions?
- Is the document complete?

- Have all requirements been identified whether functional, performance related, interface requirements, etc.?
- Have responses to both valid and invalid inputs been defined?
- Have all TBDs been resolved or addressed?
- Consistency – does this document agree with the Project Definition Document?
- Have the requirements been ranked or identified for prioritizing?

## Analysis & Design Artifacts

### Data Model

The data model describes in diagram and report format, both the logical and physical representation of the persistent data. This model should include all behavior defined in the database such as stored procedures, triggers, constraints and so forth. The model is initially developed during the Design phase and then maintained as a living document for the life of the project and beyond.

To ensure a complete data model, consider the following:

- Has the storage and retrieval of data been optimized – are the tables denormalized, all CRUD (create, read, update and delete) scenarios been considered to ensure optimal performance?
- Have indexes been defined to optimize access? Have indexes been considered for impact to other database operations?
- Have data and referential constraints been identified?

### Business Rules Document

Business rules are declaration of conditions or policy that must be met. This document provides a list of all the applicable business rules and they may be grouped for readability. These rules may be laws or code that must be followed. They must be formally expressed so they can form a basis for automation.

Consider the following when evaluating business rules:

- Are all relevant business rules listed and referred to? If a rule exists in the list but is not referred to elsewhere in the project artifacts, then the rule may not be relevant to this project?
- Are the rules defined using clear and consistent terminology to be easily understood by all project members and stakeholders?

## Object Model

The object model is used to define as well as document the design of the software system. It is a comprehensive, composite artifact encompassing all design classes, subsystems, packages, and the relationships between them.

A good object model should positively answer the following:

- Does it satisfy and support the requirements?
- Is it resistant to changes in the implementation environment?
- Is it easy to maintain in relation to other models and to system implementation?
- Is it clear how it should be implemented?
- Does it include information that is typically documented within program code? (it should not)
- Is it easily adapted to requirements changes?

## Test Artifact Set

### Test Plan

A definition of the goals and objectives of testing the developed components of the iteration or the project, the items being targeted by the testing, the approach to be taken, the resources required and the deliverables to be produced.

The plan should include tests covering data integrity, functional requirements, user interface requirements, performance and load testing, security and access control, configuration and installation.

The plan should be developed as early into the project as possible. It would be beneficial to develop the plan iteratively, adding to it as iterations are detailed. The scripts should be detailed clearly to expedite review, feedback and approval.

To ensure a concise test plan, consider the following:

- The plan clearly sets the scope of the testing by stating the stages and types of tests, features/functions to be tested, and any assumptions, contingencies, or risks that may impact the tests.
- The plan states the artifacts to be developed and/or completed, the content of those artifacts, how they will be distributed and how they will be used.
- Each requirement or feature should have at least one test or a statement explaining why the test is unnecessary.
- All requirements for testing have been identified and prioritized for each type of testing to be completed.
- All hardware, software and personnel required to complete the tests have been identified.

- The test plan includes a schedule or a list of milestones identifying major project and test related activities.

#### Test Case

This document is used to describe test cases that will be needed to ensure that the solution performs as expected. Testing will be performed for unit tests, system tests, and user acceptance tests. Documenting the test cases allows the testing process to be better planned, more rigorous, and repeatable. The test cases can be tracked throughout the testing process and can be tied back to requirements or design components if appropriate.

Test cases represent requirements that must be verified. Best practice is to develop at least two test cases for every requirement or feature. One test case would be to ensure the feature functions as defined and the other one would test abnormal or unexpected actions or data. If the test case required multiple steps, use the Test Script form instead. Each case needs to be named and steps required to be detailed with expected results documented.

#### Test Case Validation Log

This document records each test, documents each test case and records the results of the test. This form can be used for unit, integration, system and acceptance testing. This is an essential test artifact to track the testing progress and defects uncovered during the test process.

## Implementation Artifact Set

#### Deployment Plan

This artifact describes in detail the tasks required to install and test the developed product in the production environment so it may be transitioned into the user community. It includes detailed, scheduled events, persons responsible for those events/tasks, and event dependencies required to ensure a successful cutover to the new system.

If the system is replacing an existing system, conversion and migration issues must be addressed. This includes:

- migrating data from the existing system to the new,
- migration must not interrupt end user service for more than an agreed amount of time,
- the new system must be capable of paralleling the existing system,
- there must be a way to convert back to the existing system any time within the first two weeks of implementation if necessary.

The following factors should be considered when developing the deployment schedule:

- Users of the system may need to be trained.
- There may be business objectives that set the deployment date.
- There may be time periods when the demands on the business community do not allow a new system to be implemented.

- Workload peaks and other factors in existing systems and processes might prevent implementation at certain times.

#### End User Support Materials

These include any type of material that will assist users in learning, using, operating and maintaining the new system. The support materials may be documents, online help systems, computer based tutorials, etc.

## Configuration and Change Management Artifact Set

#### Change Management Plan

This document describes all configuration and change control management activities and procedures that will be performed during the course of the project lifecycle. It details the schedule of activities, responsibilities and required resources (staff, tools, computers, etc.).

#### Change Request

Changes to any project development artifacts (including the system programs) are made through a Change Request. This request must be evaluated in terms of impact to the overall design and/or requirements, effort involved to adopt the change and impact on the overall project schedule. Changes consist of enhancements to initial requirements or design as well as defects in the developed product. These requests provide a record of decisions made during the project lifecycle and ensure that all persons understand the impact a change will have on the project.

Changes are inherited during the lifecycle of any project. They are requested by all project members including the analysts, the project manager, the stakeholders, the end users, the testers, the designers, etc. These changes must be controlled to ensure the project does not get off schedule to the point implementation never occurs and to prevent changes that may impact the overall functional success of the end product. Change control also ensures that all reported defects are tracked and corrected prior to implementation.

#### Change Issue Log

This log provides the Project Manager with a way to track and manage changes and/or issues that are not part of the normal change control process. These items may include problems, exceptions, anomalies, or other items that require attention that relate to the management of the project.

## Project Management Artifacts

#### Iteration Plan

The plan is a fine grained plan for a particular project development iteration containing resources, activities and dependencies. The plan is used to schedule tasks and assign resources and to track progress against the schedule. Project team members use the plan to understand what they are assigned to do and when and what their task are dependent on, if anything. Each iteration is planned during the end of the previous iteration and includes which product features will be addressed during that iteration.

The following items (or information) must be available to define the contents of the iteration plan:

- The project plan
- Current status of the project including change requests and change issue log items
- A list of the features that need to be completed by the end of the iteration
- Any risks that must be mitigated by the end of the iteration
- A list of change requests that must be addressed (or included) in that iteration

These lists must be ranked and the plan should be aggressive so there will be items available that can be dropped based on the ranking should problems arise.

#### Risk List

This is a list of known and open risks to the project that are generally sorted in decreasing order of importance and associated with contingency or mitigation plans. This list is also the basis around which iterations are planned, with the intent to resolve or mitigate the greatest risks first. The list is created early in the project and updated throughout the lifecycle of the project. At the very least, the list should be updated at the end of every iteration.

The following are areas for consideration when identifying potential risks to a project:

- Resource Risks
  - Organization
    - Is there sufficient commitment to the project (management, testers, etc.)?
    - Is this the largest project the organization has ever undertaken?
    - Is there a well-defined systems development methodology?
  - Funding
    - Is funding in place for the project?
    - Has funding been allocated for training?
    - Are there budget limitations such that the project must be completed with a specific cost range?
    - Are cost estimates accurate?
  - People
    - Are enough people available to work on the project?
    - Do they have the right skills and experience?
    - Have they worked together before?
    - Do they believe the project can succeed?
    - Are user representatives available for reviews?
    - Are domain experts available?
  - Time

- Is the schedule realistic?
  - Can functionality be scope-managed to meet the schedule?
  - How critical is the delivery date?
  - Is there time to “do it right?”
- Technical Risks
  - Scope Risks
    - Can success be measured?
    - Is there agreement on how to measure success?
    - Are the requirements fairly stable and well understood?
    - Is the project scope firm or does it keep expanding (“scope creep”)?
    - Are project development times short and inflexible?
  - Technological Risks
    - Has the technology been proven?
    - Are the transaction volumes in the requirements reasonable?
    - Are the data volumes reasonable?
    - Are there unusual or challenging technical requirements that require the team to deal with problems they are unfamiliar with?
    - Is success dependent on new and untried products, services or technologies, new or unproven hardware, software, or techniques?
    - Are there external dependencies on interfaces to other systems? Do the required interfaces exist or must they be created?
    - Are there availability or security requirements that are extremely inflexible?
    - Are the users inexperienced with the type of system being developed?
    - Is there increased risk due to the size or complexity or newness of the technology of the system being developed?
  - External Dependency Risks
    - Does the project depend on other parallel projects?
    - Is success dependent on off the shelf products or externally developed components?
    - Is success dependent on successful integration of development tools, implementation technologies? If so, is there a backup plan for delivering the project without these tools or technologies?

#### Product Acceptance Plan

This document describes how the customer will evaluate the deliverable artifacts from a project to determine if they meet a predefined set of acceptance criteria. It details the acceptance criteria and identifies product acceptance tasks that must be completed and assigned responsibilities and required resources. This plan is created during the Project Definition phase and updated as required based on changes after iterations. The plan includes actions to be taken if the customer identifies problems.

#### Project Acceptance Report

This document will recap the project description, list any remaining enhancement requests and address any outstanding issues. This document serves as the official acknowledgement by the end user that the project has been successfully completed.

## Appendix A – Glossary

TERM / ACRONYM	DEFINITION
<b>Defect</b>	An anomaly or flaw in a delivered product. (A “bug”)
<b>Iteration</b>	A portion of the features are grouped and designed, developed, tested and implemented during one iteration. This approach is taken to enable the developers to deliver pieces of the system every 2-4 weeks instead of the customer having to wait until the entire system is built to have any portion available to them in the production environment.
<b>Architecture</b>	Refers to hardware or software, or a combination of the two that compose a particular system. Details the “pieces” that exist to make up the entire system.
<b>Prototype</b>	A working model of the proposed system. Generally a prototype only mimics the proposed system to detail the user interface requirements. There may be no connection to a real data platform, however, if there is a need to prove (proof of concept) the technology or architecture, the prototype may go to that level.
<b>Artifact</b>	Generally a document of some format but may also be a diagram or a prototype. Something that is created by a project team member for use in completing the project. Usually an artifact created in one phase of the project is used as input to a later phase.
<b>Enhancement</b>	To improve an existing system by adding new features or functionality. Enhancements are generally small in nature but grouped into one project per system or may be handled on an individual basis as requested.
<b>Deployment</b>	To distribute (implement) the developed system into the production environment to make it available for daily use by the end users.
<b>Scope</b>	The area or functionality to be covered by the project. Specifically states what will be included and what will be excluded. Clearly sets the boundaries of the development environment.