**Instructions:** For the suggested topics below, replace <text within brackets> with project-specific information. Some of the topics may not apply to all projects. In the topics where a response is optional, a choice*,* [“N/A for this report”] is included in the instructions.

**CHANGE ALL:**

**<Project Name> to your project name**

Guidelines for Selection of

Project Life Cycle Models

|  |  |
| --- | --- |
| ***Current Version:***  |  |
| ***Owner:*** |  |
| ***Date Last Updated:*** |  |
| ***Last Updated By:***  |  |
| ***Author:*** |  |
| ***Date Created:*** |  |
| ***Approved By:*** |  |
| ***Approval Date:*** |  |

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version Number** | **Date Updated** | **Revision Author** | **Brief Description of Changes** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Table of Contents**

< Generated Automatically. To update table based on changes in the document, select the table and hit F9. >

Contents

[1. Introduction 4](#_Toc497896459)

[1.1 Purpose 4](#_Toc497896460)

[1.2 Definitions 4](#_Toc497896461)

[1.3 Life Cycle Model versus Work Breakdown Structure 5](#_Toc497896462)

[1.4 Additional Reference Materials 6](#_Toc497896463)

[2. Software Development Life Cycle Models 6](#_Toc497896464)

[2.1 Waterfall Model 6](#_Toc497896465)

[2.2 Incremental 7](#_Toc497896466)

[2.3 Evolutionary (Phased) 8](#_Toc497896467)

[2.4 Spiral 9](#_Toc497896468)

[3. Selecting a Life Cycle Model for Your Project 11](#_Toc497896469)

[3.1 Why Select a Project Life Cycle Model 11](#_Toc497896470)

[3.2 Working with a Supplier’s Life Cycle Model 11](#_Toc497896471)

[3.3 Selecting the Projects Life Cycle Model 11](#_Toc497896472)

[3.4 Risk Based Selection for Life Cycle Models 13](#_Toc497896473)

[4. Applying Waterfall 13](#_Toc497896474)

[4.1 Waterfall Project 14](#_Toc497896475)

[4.2 Incremental Project 14](#_Toc497896476)

[4.2.1 Planning the Incremental Project 14](#_Toc497896477)

[4.2.2 Estimating the Incremental Project 17](#_Toc497896478)

[4.3 Evolutionary 18](#_Toc497896479)

[4.3.1 Planning the Evolutionary Project 18](#_Toc497896480)

[4.3.2 Estimating the Evolutionary Project 20](#_Toc497896481)

[4.4 Spiral 22](#_Toc497896482)

[4.4.1 Planning the Spiral Project 22](#_Toc497896483)

[4.4.2 Estimating the Spiral Project 24](#_Toc497896484)

[5. Other Considerations 25](#_Toc497896485)

[5.1 Cost and Schedule Baseline Considerations 25](#_Toc497896486)

[5.2 Financial and Budget Considerations 26](#_Toc497896487)

[6. Summary 27](#_Toc497896488)

 Guidelines for Selection of

Project Life Cycle Models

# Introduction

## Purpose

This document is intended to serve as a guide for Project Managers who need to consider different project life cycle models and tailor the established lifecycle models when planning their system development projects.

The document does not address other types of projects, such as infrastructure improvement or process improvement projects. It is also not intended to specifically apply to system maintenance activities, although, it could be useful to those planning such activities. It is also not intended to address a system’s life cycle, which includes the initial system concept and development, but also the system’s continued operations and support, its maintenance and eventual shutdown or replacement.

This Guideline document helps projects to select the Life Cycle Models based on the project’s characteristics and needs.

## Definitions

The terms “methodology” and “model” are often interchanged. However, each has a distinct definition and should not be used as synonyms.

**Methodology:** A body of *methods and rules* followed in a science or discipline *(Source: The Merriam-Webster Dictionary, 1995)*

**Life Cycle Model:** A *framework containing the processes, activities, and tasks* involved in the development, operation, and maintenance of a software product, spanning the life of the system from the definition of its requirements to the termination of its use. Also called a System Life Cycle Model. *(Source: IEEE Standard 12207.0-1996)*

For this document, this Life Cycle Model definition is further refined to focus on the project’s lifecycle.

**Project Life Cycle:** A collection of project phases, serving to *define the beginning and end of a project. (Source: PMBOK® Guide, PMI, 2000)*

There is a tendency in the information technology field to classify projects by their end product (i.e., Data Warehouse, COTS Integration, e-Commerce), but these classifications are neither life cycle models, nor methodologies. The execution of these projects follows the same methodology as any “normal” system development project, using one of the four life cycle models, with perhaps differences in how some activities are completed. For instance, the Build sub-phase of a system development project that has a data warehouse as end product could include activities such as “convert data from the legacy application.”

There is also a tendency to treat small projects as special cases. While it is true that some activities and deliverables might need to be tailored to better align with the scope of the project, projects, large or small, follow the same methodology.

One other area often confused is the difference between a model and technique. RAD (Rapid Application Development) and JAD (Joint Application Design/Development) are development techniques, not system development models. As techniques, they can be applied to any of the models.

## Life Cycle Model versus Work Breakdown Structure

A project’s life cycle model represents the framework within which the project’s activities will be completed. The projects work breakdown structure (WBS) identifies those activities and the relationships between them. Often the project’s life cycle model is shown within the WBS highest level of decomposition, as depicted in Figure 1. Note that in this example, the second digit in the WBS numbers for Build and Test activities is the increment number.

|  |  |  |  |
| --- | --- | --- | --- |
| **1 Plan**1.1 Develop Plan for Plan 1.2 Develop Process Design1.3 Develop Project Charter1.4 Estimate Effort for Define 1.5 Develop Plan for Define 1.6 Conduct Plan Tollgate**2 Define** 2.1Develop Business Requirements2.2Develop System Requirements2.3Develop Architectural Design2.4Make Build/Buy/Reuse Decision2.5Create IT Standards Bill of Material (BoM)2.6Develop Deployment Strategy2.7Conduct Bid Evaluations2.8Develop Final Project Plan2.9Conduct Define Tollgate | **Increment 1****3.1** **Build – Increment 1**3.1.1 Conduct Build Kick-Off3.1.2 Develop Detailed Design3.1.3 Develop Test Plans3.1.4 Complete System Deployment Plan3.1.5 Code and Unit Test3.1.6 Configure, Integrate and Deliver System3.1.7 Conduct Increment 1 System Testing Readiness Review**4.1** **Test** **Increment 1**4.1.1 Conduct Increment 1 System Testing | **Increment 2****3.2** **Build – Increment 2**3.2.1 Conduct Build Kick-Off3.2.2 Develop Detailed Design3.2.3 Develop Test Plans3.2.4 Complete System Deployment Plan3.2.5 Code and Unit Test3.2.6 Configure, Integrate and Deliver System3.2.7 Conduct System Testing Readiness Review3.2.8 Conduct Build Tollgate**4.2** **Test – Increment 2**4.2.1 Conduct Increment 2 System Testing4.2.2 Conduct Increment 1 Regression Testing4.2.3 Conduct System testing for Integrated System.4.2.4 Prepare Pre-Production Environment4.2.5 Conduct Acceptance Testing Readiness Review4.2.6 Conduct Acceptance Testing4.2.7 Analyze Defects and Change Requests4.2.8 Verify Production Readiness4.2.9 Conduct Test Tollgate | **5 Deploy** 5.1 Migrate to New System5.2 Update Site Deployment Materials and System Documentation5.3 Transition to Operations5.4 De-Install Legacy System5.5 Close Project |

Figure 1 - Sample WBS within an Incremental Life Cycle Model

## Additional Reference Materials

There are some sources available to the Project Manager who wants to learn more about the models presented in this paper. References used in this paper include:

* The Project Manager’s Guide to Software Engineering’s Best Practices, Mark J. Christensen and Richard H. Thayer, IEEE Computer Society, 2001
* Software Engineering: A Practitioner’s Approach, Roger S. Pressman, McGraw-Hill, 1992
* Dynamics of Software Development, Jim McCarthy, Microsoft Press, 1995
* Managing Software Development Projects, Neal Whitten, John Wiley and Sons, 1995
* Software Project Management: A Unified Framework, Walker Royce, Addison-Wesley, 1998

# Software Development Life Cycle Models

The following types of project Life Cycle Models for projects to select from and manage their projects:

* Waterfall
* Incremental
* Evolutionary
* Spiral

The following sections elaborate on each of these models.

## Waterfall Model

The Waterfall Life Cycle Model is the traditional approach to system development projects and is characterized by sequential phases. The phases follow one another, sequentially, from project initiation to implementation of project deliverables into a production environment. This approach is also known as “Big Bang” development; all requirements, all design, all deliverables are contained in one release

A modification to this model, referenced as the Modified Waterfall, recognizes there is a benefit to overlapping the phases to some degree, by starting the Detailed Design Sub Phase while finishing the Define Phase, for instance.

Figure 2 below depicts a typical project following this model.

Figure 2 - Traditional Waterfall Life Cycle Model

1The life cycle models shown are accepted industry models for software development projects. References canbe found in The Project Manager’s Guide to Software Engineering’s Best Practices, written by Mark J. Christensen andRichard H. Thayer and published by IEEE Computer Society in 2001.

## Incremental

As its name implies, the Incremental Life Cycle Model supports the development of the system in increments. In the Incremental Model, the project begins with the Plan and Define, which concludes when all requirements, both technical and business, are identified and allocated to specific increments of the system.

The initial increment typically includes those software modules that address the core requirements and their functions. Subsequent increments introduce additional functions. Each increment then follows either the Traditional Waterfall Model or the Modified Waterfall Model for the Build, Test and, optionally, Deploy activities.

Note that not every increment need be deployed into the production environment upon completion. Figure 3 depicts this delayed deployment decision. Increment-1 does not get deployed until Increment-2 is ready for deployment. This could be based on the determination that the functions developed in Increment-1 need some of the functions in Increment-2 before there is a business value, and that Increment-1 has sufficient technical risk in it that the IT Team felt early development and testing of it was required.



Figure 3 - Typical Incremental Life Cycle Model

## Evolutionary (Phased)

The Evolutionary Life Cycle Model also supports the development of a system in increments. However, it differs from the Incremental Model in that the requirements are developed as part of each increment. This model requires that the project scope activity be thorough, resulting in a solid understanding of the high-level functions to be implemented.

Each of those high-level functions is then allocated to one of the increments for further definition and implementation.

In the Evolutionary Model, the initial increment typically includes only those requirements that address the core requirements and their functions with subsequent increments introducing additional functions. Each increment then follows either the Traditional Waterfall Model or the Modified Waterfall Model for the Define, Build, Test and, optionally, Deploy activities. As with the Incremental Model, not every increment need be deployed into the production environment, as shown in Figure 4 below.



Figure 4 - Typical Evolutionary Life Cycle Model

## Spiral

In 1988, Dr. Barry Boehm introduced the concept of a spiral life cycle. Emphasizing the use of repeated prototyping, the Spiral Model was developed to address some of the risks associated with system development with new technology or in new application areas. The focus of early prototype spirals is to expose risks and “tease out” requirements from the technology or the business community. Examples of the risks that might be examined with the prototypes include network capacity, ineffective application user interface interaction, or poorly defined business needs.

The approach used with the Spiral Model includes developing enough of an understanding of the requirements to initiate the development activities and to field something for user feedback and requirements clarification or confirmation. The lessons learned from each spiral are then fed into the next loop. Often, several iterations of the spiral are needed before a real understanding of the requirements is achieved, or before the technical risks associated with the project are fully understood.

It is important to realize that, when using a spiral approach, the software developed in support of the prototypes should be viewed as “managed throw-away” code. That is, the code should be managed through the team’s configuration management practices, and a certain amount of quality control should be applied. If there is an interest in leveraging those prototype components in the final production system, then the software should be subjected to the same degree of rigor with regards to code reviews, standards compliance, testing, and change control.



Figure 5 - Typical Spiral Life Cycle Model

Use of the Rapid Application Development technique (RAD) in these early prototypes is often applicable. However, if this technique is chosen, it is important that the Project Team have a clear goal in mind, with well-defined requirements for that particular prototype. Also, the “user” being served by the prototype must be engaged. By “user,” we mean either the technical subject matter expert who is concerned about how a certain technology might respond, or the business subject matter expert with concerns about a particular business function.

# Selecting a Life Cycle Model for Your Project

## Why Select a Project Life Cycle Model

Most project life cycles share certain common characteristics:

* They typically start with lower costs and staffing requirements, grow towards maximum levels of these in the middle-to-end of the project, and drop quickly near completion.
* The probability of completing the project is lowest at the start of the project when the level of risk and uncertainty are at their highest. The probability of success increases throughout the project as requirements are defined and performance validated, thereby decreasing the level of risk and uncertainty.
* The ability of the Stakeholders to influence the project is greatest at the beginning phases of a project.

Taking the above characteristics into account, the Project Manager, working with the Project Team and the Business Sponsor, should document a project-appropriate life cycle including phases, activities, deliverables, roles, and checkpoints. This project life cycle then serves as the framework for organizing and planning the project and helping the Project Manager manage the triple constraints of scope, time, and costs.

The models discussed earlier in this paper are typical project life cycle models that should be considered. When planning a project, the Project Manager, working with the Project Team and Business Sponsor, should deliberately select the life cycle model to be applied to the project. This selection, to be performed at the high level as part of the project strategy development during the Plan, should be revisited and revalidated during the detailed project planning completed in the Define.

## Working with a Supplier’s Life Cycle Model

Projects involving outside suppliers are performing the application development activities for the Build on behalf of the organization. In these situations, the Supplier’s chosen life cycle model may influence the model chosen for the overall project.

It is important when evaluating a proposal from a Supplier to ask about the life cycle model they plan to apply and to understand why that particular model was selected, especially if the proposed model differs from the selection made by the organization. A difference in model selection is something that needs to be discussed by the Project Manager and the potential Supplier; it could be an indication of a misunderstanding of the project’s objectives or approach.

The Project Manager will need to integrate the Supplier’s model into the project’s overall life cycle model, bearing in mind that the Supplier is performing activities and providing deliverables for part of the project’s work breakdown structure. As a general rule of thumb, if the Supplier is proposing to use one of the iterative approaches (incremental, evolutionary or spiral), then the Project Manager should plan for more reviews of Supplier deliverables.

## Selecting the Projects Life Cycle Model

During the strategy development activity (part of “Develop the Project Charter,” Plan), the Project Team needs to select a project life cycle model. There are some factors the Project Team should consider when making that selection.

While Table 1 identifies some of the factors the team should consider, it is important for the Project Manager to realize:

* The list is not comprehensive. The Project Team also needs to consider the project’s constraints, risk factors, history, and any other knowledge they possess about the project and its environment.
* The list is not a checklist. It should be used to trigger thoughts and discussions, not to prescribe one life cycle model over another. The Project Team should consider seeking the expertise of staff within the PMO if one model cannot be selected over another.

As part of the development of the Project Plan (Plan), the team should validate the selection performed as part of the project strategy development. Note that as additional project knowledge is gained, it is possible that risk factors may change and that knowledge of user expectations may increase. These changes may warrant another life cycle model selection.

| If your project has this characteristic, | Then the selection of this life cycle model would be: |
| --- | --- |
| Waterfall | Incremental  | Evolutionary | Spiral |
| Unclear Requirements | Poor | Good | Excellent | Excellent |
| New Technology or Architecture Involved | Poor | Fair |  | Excellent |
| Risk Management Important | Fair | Good |  | Excellent |
| Reliable System Required | Good | Good | Excellent | Excellent |
| Schedule Predictability Required (mandated delivery dates) |  | Good | Excellent | Fair |
| Progress Visibility Required | Fair | Good | Good | Good |
| Cost Predictability Required | Excellent | Good | Excellent | Fair |
| Product Scalability Required (application is core to a larger system) | Excellent | Good | Good | Excellent |
| Support Midcourse Corrections | Fair | Excellent | Excellent | Excellent |
| Low User Involvement | Excellent | Fair | Poor | Poor |
| Multiple Vendors | Good | Fair | Fair | Poor |
| Anticipated long duration (more than 12 months) | Poor | Excellent | Excellent | Good |
| General Concern | “Big Bang” could be an explosion | Requires strong, skilled management | Requires strong, skilled management | Not easily understood by business partners |

Table 1 – Life Cycle Model Selection Considerations2

2 The values contained in the table are provided purely as guidelines, not prescribed selections. They have been developed based on reviewing the materials referenced in Section 1.4 of this document and on the experiences of multiple IT project managers representing more than 100 man-years experience with projects applying these models.

***Works with the poorly understood requirements*** refers to how well the life cycle model works when you or your customer poorly understands the systems’ requirements or when your customer is prone to change requirements. It indicates how well the model is suited to exploratory software development.

***Works with poorly understood architecture*** refers to how well the life cycle model works when you are developing in a new application area, or when you are developing in a familiar applications area but are developing unfamiliar capabilities.

***Produces highly reliable system,*** refers to how many defects a system developed with the life cycle model is likely to have when putting into operation.

***Produces system with large growth envelope,*** refers to how easily you are likely to be able to modify the system in size and diversity over its lifetime. This includes modifying the system in ways that were not anticipated by the original designers.

***Manage Risks,*** refers to the model’s support for identifying and controlling risks to the schedule, risks to the product and other risks.

***Can be constrained to a predefined schedule,*** refers to how well the life cycle model supports delivery of software by an immovable drop-dead date.

***Has low overhead,*** refers to the amount of management and technical overhead required to use the model effectively. Overhead includes planning, status tracking, document production, package acquisition and other activities that aren’t directly involved in producing software itself.

***Allows for midcourse corrections,*** refers to the ability to change significant aspects of the product midway through the development schedule. This does not include changing the product’s basic mission but does include significantly extending it.

***Provides customer with progress visibility,*** refers to the extent to which the model automatically generates signs of progress that the customer can use to track the project’s status.

***Provides management with progress visibility,*** refers to the extent to which the model automatically generates signs of progress that the management can use to track the project’s status.

***Requires little manager or developer sophistication,*** refers to the level of education and training that you need to use the model successfully. That includes the level of sophistication you need to track progress using the model, to avoid risks inherent in the model, to avoid wasting of time using the model, and to realize the benefits that led you to use the model in the first place.

## Risk-Based Selection of Life Cycle Models

This classification can also be one of the indicators as to the life cycle model the Project Team should have selected. It is reasonable to expect that high-risk projects may use the Spiral Life Cycle Model, which allows the Project Team to address some of the risks that have resulted in this classification. Low-risk projects will usually be completed using the Waterfall Life Cycle Model. Project teams working the medium risk projects will usually select one of the iterative models, based on the factors discussed earlier in these guidelines.

All Project Managers assigned to low-risk projects shall establish a baseline at the conclusion of the Define. This baseline will be used to monitor project performance.

# Applying Waterfall

|  | WATERFALL | INCREMENTAL | EVOLUTIONARY | SPIRAL |
| --- | --- | --- | --- | --- |
| **PLAN** | Occurs once | Occurs once | Occurs at project inception and beginning of each increment | Extensive planning  |
| Validated at the end of each life cycle phase | Validated at the end of each life cycle phase and end of increment | Validated at the end of each life cycle phase and the end of increment | Extended risk analysis |
|  | Includes high level requirements definition | Includes high level requirements definition | Includes high level requirements definition |
|  |  | Allocates requirements to increments | Allocates risks to prototypes |
| **DEFINE** | Occurs once | Occurs once | Occurs in each increment | Includes a number of prototypes |
|  | Addresses full project | Addresses just that increment | Called “exploration stage.” |
|  | Allocates specific requirements to increments | Validates requirements allocation | Requires extended period of time |
|  |  |  | Select life cycle model for core project |
| **BUILD** | Occurs once | Occurs in each increment | Occurs in each increment | Follows practice of chosen life cycle model |
| Single design document | Multiple logical design documents | Multiple logical design documents |  |
| Change controls within current Buildsub-phase | Change control across increments during Buildsub-phase | Changes control across increments during Buildsub-phase | Follows practice of chosen life cycle model for Build sub-phase |
| **TEST** | Single test cycle | Multiple test cycles | Multiple test cycles |  |
|  | Regression testing | Regression testing |  |
| **DEPLOY** | Single release | Multiple releases | Multiple releases | Follows practice of chosen life cycle model |
| Project Close occurs Once | Close occurs in each increment and finally at the end of project | Close occurs in each increment and finally at the end of project | Follows practice of chosen life cycle model |

Table 2 – Waterfall Project Life Cycle Comparison Chart

## Waterfall Project

since the waterfall lifecycle model is most commonly used this guideline will not provide further guidance on tailoring the methodology to support that project life cycle model.

## Incremental Project

One of the benefits of the Incremental Life Cycle Model is that it allows a project to be managed in small pieces. This supports the ability to end a project at logical termination points in its life cycle, as opposed to during a development activity.

### Planning the Incremental Project

Table 3 depicts the mapping of the Waterfall Model to an Incremental Project Life Cycle Model. The bold, italicized words in the table highlight the differences between the Waterfall Model and Incremental Model. Several key points are identified in that Table:

* Initial project phases do not change.
* Subsequent phases are replicated for each increment.
* Not every increment need be deployed.

#### Plan Impacts

It is important to realize when planning an incremental project that all requirements, technical and business, need to be identified and understood before starting Increment Build. This understanding is needed to ensure that “natural” increments are identified. Therefore, when planning the Define in this model, emphasize activities associated with the review of requirements with business and technical subject matter experts, and plan for more requirement reviews during the Define.

This model requires that the business actively participate not just at the beginning and end of the project, but throughout the project, as increments are made available for User Acceptance Testing. Obtaining that business resource commitment is an important activity that must be accomplished when planning the incremental life cycle. It cannot be skipped. Also, the Business Sponsor needs to have a clear understanding of the deliverables to be provided (or not provided) as part of each increment. This includes an understanding of which increments will be taken through deployment and which ones will not. This need is best addressed by including the Business Sponsor (or representatives) in the determination of the increments and the development of each increment’s detailed work breakdown structure.

Initially, the planning for each of the increments should be done at a sufficient level of detail to know:

* Where interdependencies between requirements exist,
* When user commitments are required, and
* When certain long-term activities, such as hardware requisitions, need to be initiated.

The more detailed planning should then be accomplished at the initiation of each increment, leveraging the lessons learned from earlier increments and factoring in any change controls that might have been identified.

When planning a project applying the Incremental Model, the Project Manager needs to pay extra attention to data and configuration management. The Project Plan needs to define how the deliverable documents (i.e., system design and test documents) will be developed, managed and consolidated across the various increments.

#### Define Impacts

As mentioned earlier, it is important that all requirements, technical and business, be identified and understood before Increment 1’s Build starts. This means the Define in this model will probably be of a longer duration, with more requirements reviews by business and technical subject matter experts.

#### Build Impacts

The Build is repeated for each increment, building upon the design from the previous increment. Before beginning the design activities, the Project Team should re-validate the Project Plan, schedule and budget, especially for the increment about to begin, to ensure that lessons learned from previous increments have been considered and that outstanding Change Requests can be accommodated. If there are issues that indicate the increment will not happen as planned, the Project Manager should notify all Stakeholders and schedule a Project Review to address the issues.

#### Test Impacts

Note that each increment will require a test plan be developed for that increment’s deliverables, plus a regression test plan addressing the functions delivered in the earlier increments. So, the test period associated with the second and subsequent increments will usually be longer than the period associated with earlier increments. Also, configuration management complexity will increase as additional functions are implemented.

As mentioned earlier, change control is particularly important when using the Incremental Life Cycle Model. Changes in requirements or identification of additional requirements at a later date in the project’s evolution could result in the need for significant rework of an already completed increment. This rework not only affects the code that has been developed but potentially the associated design documents, the increment’s test plan, and the overall system test plan. The change control procedures should state how this rework will be addressed, who will authorize it, how it will be integrated into later increments and how impacts will be communicated to all Stakeholders.

#### Deploy Impacts

There could be multiple deployment phases in a project using the Incremental Life Cycle Model. The Project Manager needs to consider the timing of those deployments, factoring in the impact to the business and operations community. The deployments should be staged far enough apart to allow the users to be somewhat comfortable with the previous increment before releasing another increment. A general rule of thumb is one release per quarter. The Project Team should use release notes to highlight the features contained in the new increment to the user and operations community.

**Incremental Life Cycle Model**



Figure 8 - Incremental Life Cycle Model

| Phases | Project Deliverables |
| --- | --- |
| **Plan** - The purpose of the Plan is to identify and documents the project's plans, including the strategy, charter, schedule and preliminary budget. | * Project Charter (PC)
* Preliminary Project Plan & Schedule [PP(P)]
 |  |
| **Define** - The purpose of the Define is to refine the business, user and functional requirements to develop a clear understanding of software requirements and ***to establish a refined schedule and budget for each increment. The original Project Plan should be reviewed and details associated with the applicable increment revalidated.*** | * Systems Requirements Specifications (SRS)
* Architectural Design (AD)
* Project Schedule/Budget
* IT Standard BOM (BOM)
* Product Quality Plan (PQP)
* Business Requirements (BRD)
* Requirements Traceability Matrix (RTM)
* Final Project Plan [PP(F)]
 | * IT supplier service Agreements.
* Project Plan & Schedule.
 |
| **Build** - The purpose of the Build is to develop a precise statement of exactly ***what a specific increment of the application is to do and how it is to do it.*** | * System Design Document (SDD)
* User Acceptance Test Plan (UATP)
* System Deployment Plan (DP)
* Source Code (SC)
* System Test Plan (STP)
* System Documentation
* Operations Maintenance and Support Plan (OSP)
* System Test Summary Report (STSR)
 | **For Each Increment*** Training Curriculum
* Design Documents
* Test cases
* Revise System Deployment plan
* Revise Operations & support Plan
 |
| **Test** - The purpose of Test is to validate the system by Conducting System Testing and Acceptance Testing, ***including regression testing against prior increments. Documentation associated with this increment’s function is also developed during this phase.*** | * User Acceptance Test Summary Report (UATSR)
* Pilot Deployment Plan (PDP)
 | **For Each Increment*** Coded/Configured System
* Training Material
* Test Results
* System & User Documentation
* Revise Operations & Support plan
 |
| **Deploy** - The purpose of the Deployment is to package, release, install, configure and update a software system. It includes activities related to user and application support training. | * Site Deployment Plan (SDP)
 | **For Each Increment*** Training
* Deployed System after Customer Acceptance
 |

Table 3- Incremental Life Cycle Model Mapped to Waterfall

### Estimating the Incremental Project

Estimating a project using the Increment Life Cycle Model is similar to estimating a project using the Waterfall Life Cycle Model. Emphasis should be placed on activities associated with the review of requirements with business and technical subject matter experts during the Define. Therefore, the estimate for effort associated with the Define should be higher than in a project using the Waterfall Life Cycle Model.

There should also be additional effort allocated to testing. Each increment developed will need to be tested on its own, integrated with earlier increments and subsequently subjected to regression testing. Each increment will have that many more test cases to be processed and reviewed, thereby increasing the required effort.

The incremental development also impacts configuration management and change control efforts. Elements under development require configuration management as do those already developed or pending deployment.

Each increment should be estimated on its merit. Using the estimating templates, the Project Manager should develop an estimating worksheet for each increment. The Project Manager should then prepare a summary estimate worksheet that represents the totals for each increment as part of the project planning activities.

These estimates should be re-validated and updated as needed at the conclusion of each increment, before the initiation of the next increment. Significant deviations from earlier estimates should be discussed as part of the Tollgate Review, and a determination made about how to proceed.

## Evolutionary

The Evolutionary Life Cycle Model is another model that allows a project to be managed in small pieces. It differs from the Incremental Life Cycle Model in that the requirements activities are aligned with each increment. A project following this model is often thought of as a program with multiple related projects executed sequentially.

### Planning the Evolutionary Project

Planning an Evolutionary project is similar to the planning performed for an Incremental project. Key thoughts to bear in mind when planning for this type of project include:

* The Plan (or planning) of the project does not change.
* Subsequent phases are replicated for each increment.
* Not every increment need be deployed.

**Table 4** depicts the mapping of the Waterfall model to an Evolutionary Project Life Cycle Model. The bold, italicized words highlight the differences between the Waterfall model and the Evolutionary model. Note that the Define is performed for each increment, as opposed to once at the beginning of the project.

#### Plan Impacts

It is important to realize when planning an Evolutionary project that a high-level understanding of the requirements, both technical and business, is needed, to support the allocation of different releases of the application. This understanding is needed to ensure that “natural” increments are identified, and that priority requirements are addressed in the earlier increments. Therefore, when planning the Plan in this model, emphasize activities associated with the determination of requirements with business and technical subject matter experts, expanding the activities associated with project scope development.

An evolutionary approach to a project also requires that the business actively participate not just at the beginning and end of the project, but throughout the project, as the requirements and design associated with each increment are expanded and as increments are made available for User Acceptance Testing. Obtaining that business resource commitment is an important activity that must be accomplished when planning the Evolutionary project. It cannot be skipped. Also, the Business Sponsor needs to have a clear understanding of the deliverables to be provided (or not provided) as part of each increment. This includes an understanding of which increments will be taken through deployment and which ones will not. This need is best addressed by including the Business Sponsor (or representatives) in the determination of the increments and the development of each increment’s detailed work breakdown structure.

Initially, the planning for each of the increments should be done at a sufficient level of detail to know:

* Where interdependencies between requirements exist,
* When user commitments are required and
* When certain long-term activities, such as hardware requisitions, need to be initiated.

The more detailed planning should then be accomplished at the initiation of each increment, leveraging the lessons learned from earlier increments and factoring in any change controls that might have been identified.

As in the Incremental Model, the Project Manager needs to pay extra attention to data and configuration management. The Project Plan needs to define how the deliverable documents (i.e., system design and test documents) will be developed, managed and consolidated across the various increments.

#### Define Impacts

Each increment is initiated with a Define, which builds on the requirements analysis accomplished in the earlier increments. The Project Team should leverage the lessons learned from earlier increments and factor in any change controls that might have been identified and allocated to the increment under analysis.

Before beginning the define activities, the Project Team should re-validate the Project Plan, schedule, and budget, especially for the increment about to begin, to ensure that lessons learned from previous increments have been considered and that outstanding Change Requests can be accommodated. If there are issues that indicate the increment will not happen as planned, the Project Manager should notify all Stakeholders and schedule a Project Review to address the issues.

The new requirements should be documented in a manner that allows their relationship to the new increment to be easily identified and yet supports the desire to have a single requirements document for the application. One recommended technique is to establish a separate chapter or section in the requirements document for each increment. Another technique is to include the increment name as part of the requirements identification scheme.

#### Build Impacts

The Detailed Design Sub-Phase is repeated for each increment, building upon the design from the previous increment. It is recommended that the design documents for each increment be treated as chapters, or sections, of the overall application design document. This will result in a single design document to be placed under configuration management, as opposed to multiples. The same consideration needs to be applied to the system’s test documents, user documents, and other non-software deliverables.

#### Test Impacts

Note that each increment will require a test plan be developed for that increment’s deliverables, plus a regression test plan addressing the functions delivered in the earlier increments. So, the test period associated with the second and subsequent increments will be longer than the period associated with earlier increments. Also, configuration management complexity will increase as additional functions are implemented.

Change control is particularly important when using the Evolutionary Life Cycle Model. Changes in requirements or identification of additional requirements at a later date in the project’s evolution could result in the need for significant rework of an already completed increment. This rework not only affects the code that has been developed but potentially the associated design documents, the increment’s test plan, and the overall system test plan. The change control procedures should state how this rework will be addressed, who will authorize it, how it will be integrated into later increments and how impacts will be communicated to all Stakeholders.

#### Deploy Impacts

There will be multiple deployment phases in a project using the Evolutionary Life Cycle Model. The Project Manager needs to consider the timing of those deployments, factoring in the impact to the business and operations community. The deployments should be staged far enough apart to allow the users to be somewhat comfortable with the previous increment before releasing another increment. A general rule of thumb is one release per quarter. The Project Team should use release notes to highlight the features contained in the new increment to the user and operations community. Consideration should also be given to the type of training that might be required to support that new increment.

### Estimating the Evolutionary Project

Estimating a project using the Evolutionary Life Cycle Model is similar to estimating a project using the Incremental Life Cycle Model. Emphasis should be placed on activities during the Plan associated with the definition of the project scope and the associated high-level requirements with business and technical subject matter experts. It is during these activities that the number of increments and deployment decisions will be made. Consequently, the effort estimate associated with the Plan should be higher than in a project using either the Waterfall or Incremental Life Cycle Models.

This model also requires additional effort be allocated to other activities. One of those activities is testing. Each increment developed will need to be tested on its own, integrated with earlier increments and subsequently subjected to regression testing. Each increment will have that many more test cases to be processed and reviewed, thereby increasing the required effort. As with the Incremental Model, development with the Evolutionary Model increases the effort associated with configuration management and change control.

Each increment should be estimated on its merit. Using the estimating templates, the Project Manager should develop an estimating worksheet for each increment. The Project Manager should then prepare a summary estimate worksheet that represents the totals for each increment as part of the project planning activities.

These estimates should be re-validated and updated as needed at the conclusion of each increment, before the initiation of the next increment. Significant deviations from earlier estimates should be discussed as part of the Tollgate Review, and a determination made about how to proceed.

**Evolutionary Life Cycle Model**



Figure 9- Evolutionary Life Cycle Model

| Phases | Project Deliverables |
| --- | --- |
| **Plan** - The purpose of the Plan is to identify and documents the project's plans, including the strategy, charter, schedule and preliminary budget. | * Project Charter (PC)
* Preliminary Project Plan & Schedule [PP(P)]
 |  |
| **Define** - The purpose of the Define is to refine the business, user and functional requirements to develop a clear understanding of software requirements and ***to establish a refined schedule and budget for each increment. The original project plan should be reviewed and details associated with the applicable increment revalidated.*** | * Systems Architecture Document
* IT Supplier Service Agreements
* Project Schedule/Budget
* IT Standard BOM (BOM)
* Product Quality Plan (PQP)
* Business Requirements (BRD)
* Requirements Traceability Matrix (RTM)
* Final Project Plan [PP(F)]
 | For Each Increment:* System Requirements Document Agreements.
* Revise System Deployment Plan
* Revise Operations Maintenance & support Plan.
 |
| **Build** - The purpose of the Build is to develop a precise statement of exactly ***what a specific increment of the application is to do and how it is to do it.*** | * System Design Document (SDD)
* User Acceptance Test Plan (UATP)
* System Deployment Plan (DP)
* Source Code (SC)
* System Test Plan (STP)
* System Documentation
* Operations Maintenance and Support Plan (OSP)
* System Test Summary Report (STSR)
 | **For Each Increment*** Training Curriculum
* Design Documents
* Add Test cases
* Revise Test Plan
* Revise System Deployment plan
* Revise Operations Maintenance & support Plan
 |
| **Test** - The purpose of the Test is to validate the system by conducting System Testing and Acceptance Testing, ***including regression testing against prior increments. Documentation associated with this increment’s function is also developed during this phase.*** | * User Acceptance Test Summary Report (UATSR)
* Pilot Deployment Plan (PDP)
 | **For Each Increment*** Coded/Configured System
* Training Material
* Test Results
* System & User Documentation
* Revise Operations Maintenance & Support plan
 |
| **Deploy** - The purpose of Deploy is to package, release, install, configure and update a software system. It includes activities related to user and application support training. | * Site Deployment Plan (SDP)
 | **For Each Increment*** Training
* Deployed System after Customer Acceptance.
 |

**Table 4 - Evolutionary Life Cycle Model Mapped to Waterfall**

## Spiral

### Planning the Spiral Project

Planning a project using the Spiral Life Cycle Model is a complex undertaking. Of the four life cycle models discussed, the Spiral Model is the one that requires experienced Project Managers and engaged Business Sponsors. In addition to planning and managing the actual development project, the Project Manager needs to plan and manage some number of prototype projects associated with the core project. Table 5 depicts the mapping of the Waterfall Model to a Spiral Project Life Cycle Model. The bold, italicized words highlight the differences between the Waterfall Model and the Spiral Model.

**Plan Impacts**

The determination to use this model should have been made based on a need to address risks identified during the project scope definition. Therefore, the planning activities usually include an in-depth risk analysis activity to determine which of the risks can be addressed by the use of prototypes, to identify the numbers and complexities of those prototypes and the logical sequence of when the prototypes should be undertaken.

The Project Manager should begin the more detailed project planning by planning the prototype development efforts required to clarify the “fuzzy” requirements, again addressing both technical and business requirements. When planning the prototype iterations, often called the “exploration stage” of the project, it is important to remember that, while the code developed is viewed as “throw-away,” some of it might be used in the final solution. Therefore, disciplines such as configuration management, testing, and documentation are still required.

When planning the prototype activities, the Project Manager needs to have a defined goal in mind, to know when the exploration stage will end and the actual system development project will begin. It is not unheard of for projects to get “locked” into the exploration stage, with the resulting prototype software becoming the production system.

Change control is particularly important when using the Spiral Life Cycle Model. Consideration needs to be given to change control management during the exploration stage when the prototypes are under development. Again, because prototype software is often viewed as expendable, the application of change control discipline is often set aside. This could result in chaos, with the inability to resurrect an earlier version of a prototype for use in the production solutions. The Project Manager needs to prevent this tendency.

As the exploration stage progresses and more prototypes are developed, changes in requirements or identification of additional requirements at a later date need to be measured against earlier prototypes, to ensure the determinations made with those earlier prototypes still hold true. For instance, if a prototype was used to determine the sufficiency of a networks’ capacity, a significant change in the transaction set could invalidate the prototype results.

And, when the project is in its actual development stage, changes in requirements could result in the need for significant rework of an already completed increment. This rework not only affects the code that has been developed but potentially the associated design documents, the increment’s test plan and the overall system test plan. The change control procedures should state how this rework will be addressed, who will authorize it, how it will be integrated into later increments and how impacts will be communicated to all Stakeholders.

**Define and Build Impacts**

The Define of a project using the Spiral Model is typically much longer in duration and requires much more effort than in the other models. It is during these phases that many of the prototypes will be specified, developed and tested. Hence, it is sometimes referenced as the exploration stage of the project.

The Spiral Model requires active involvement by the Business Sponsors during these activities. The prototypes that are developed address, not just the technical requirements but also the business requirements. Consequently, the business community needs to participate in the definition of the prototypes and the validation of the prototype results. It is also possible that one or more of the prototypes is desired to investigate a new business process, such as the introduction of a new call center feature, thereby requiring business participation.

**Post-Build Phase Impacts**

The actual development project begins upon conclusion of this exploration stage. At that time, the Project Team needs to decide which of the other life cycle models it is going to apply in the development of the production system. The Define and Build associated with that development activity is usually very short, consisting of a re-validation of the requirements addressed during the earlier project-level Define, the exploration stage.

### Estimating the Spiral Project

Estimating a Spiral project requires experience. The Project Manager and the Business Sponsors will need to be able to determine the payback of each prototype regarding cost of developing the prototype versus the cost associated with the risk being investigated. This requires that an estimate is developed for each prototype, in addition to the estimate for the actual development project. Industry best practice in this area is to estimate and fund the exploration stage as an independent project, and then use the lessons learned from it to estimate the actual development project.

It should be expected that a project using the Spiral Model is more expensive up front than projects conducted using other models. The trade-off that is being made is a determination of, and removal of, uncertainties in the project’s requirements. The Define and Detailed Design Sub-Phases in a Spiral project is much longer in duration and more effort-intense than in projects using the other models. There is higher overhead associated with the management of the project, including costs of configuration management, quality management, and general project management. There might also be additional hardware costs to be considered as the Project Team experiments with different platforms and configurations

Each prototype activity should be estimated on its merit. Using the estimating templates, the Project Manager should develop an estimating worksheet for each one. The Project Manager should then prepare a summary estimate worksheet that represents the totals for each spiral, as part of the project planning activities. A separate set of estimating worksheets should then be developed, and summarized, addressing the production system development activities.

These estimates should be re-validated and updated as needed at the conclusion of each spiral iteration, before the initiation of the next spiral. Significant deviations from earlier estimates should be discussed as part of the Tollgate Review, and a determination made about how to proceed.

**Spiral Life Cycle Model**

| Phases | Project Deliverables |
| --- | --- |
| **Plan** - The purpose of the Plan is to identify and document potential concepts to satisfy a defined business need.Each prototype is planned with a specific goal identified. Change control processes governing the prototypes are developed. The processes by which the prototype results will be communicated and associated decisions made are documented. | * Project Plan & Schedule
 | * System Deployment plan
* Operations Maintenance and Support Plan.
 |
| **Define** - The purpose of the Define is to identify system requirements. Both the technical and non-technical risks are assessed. Technical risks are addressed by developing a proof of concept that eliminates the risk either by proving the feasibility of any solution in a major risk area or by providing a satisfactory work around. Benefits are direct business benefits and technical benefits. | * Software Architecture Document.
* Project Schedule/Budget
* IT Supplier Services Agreements
* Project Plan and Schedule
* Risk Analysis Reports
 | * Test Plans
* System Requirements Document.
* System Deployment Plan
* Training Requirements
* Operations Maintenance & Support Plan
 |
| **Build** - The purpose of the Build is to develop a precise statement of exactly ***what the application is to do, how it is to do it and building the application*** | * Design Documents
* Test Cases
 | * System Deployment plan
* Operations Maintenance & support Plan
 |
| **Test** - The purpose of the Test is to validate the system by Conducting System Testing and Acceptance Testing, Documentation associated with the function is developed during this phase. | Dependent on Life Cycle Model was chosen for actual development |
| **Deploy** - The purpose of Deploy is to package, release, install, configure and update a software system. It includes activities related to user and application support training. | Dependent on Life Cycle Model was chosen for actual development |

Table 5 - Spiral Life Cycle Model

# Other Considerations

## Cost and Schedule Baseline Considerations

The policy identifies distinct points in a project’s life cycle where the project’s cost and schedule are baselined for tracking purposes. This policy states that all projects will be classified, during the Define, as falling into one of the three risk categories depicted in Figure 6. The CIO organization will determine this classification, which will be recorded with the baseline within the Project Tracking System (PTS).

This classification can also be one of the indicators as to the life cycle model the Project Team should have selected. It is reasonable to expect that high-risk projects may use the Spiral Life Cycle Model, which allows the Project Team to address some of the risks that have resulted in this classification. Low-risk projects will usually be completed using the Waterfall Life Cycle Model. Project Teams working the medium risk projects will usually select one of the iterative models, based on the factors discussed earlier in these guidelines.



Figure 10 - Project Classifications for Baselines

The policy then states that all Project Managers assigned to low-risk projects will establish a baseline at the conclusion of the Define. This baseline will be used to monitor project performance.

The policy also states that all Project Managers will establish an initial project schedule and budget estimate at the conclusion of the Plan for each of their projects. For projects using the Evolutionary Life Cycle Model, the project performance baseline should be established at the conclusion of the Plan for the initial increment, with estimates for all increments based on knowledge at hand.

All Project Managers assigned to medium and high-risk projects will provide an updated schedule, and budget estimates at the conclusion of a project’s Define and will establish a baseline at the conclusion of the Build. Project Managers applying the Spiral Life Cycle Model should revisit this baseline as part of the analysis of the results from each iteration. The baseline should be modified as needed to reflect the newly gained knowledge. Project Managers using one of the iterative approaches (either incremental or evolutionary), should re-validate the baseline as they initiate each subsequent increment, using as inputs the experience gained on the completed increment.

## Financial and Budget Considerations

When a project budget is initially requested, it is often a requirement for the Project Manager to provide an anticipated *total* project cost and some projection as to when those costs would incur. This means the Project Manager needs to have completed estimation to some degree of detail: that a life cycle model has been considered, the numbers and scope of prototypes have been identified, that the number of increments is known.

Unfortunately, that knowledge is often not gained until the conclusion of the project’s Plan. And, in the case of a project that is using the Spiral Model, because of risks associated with the project, that knowledge is not known until the conclusion of the prototyping activities.

That is why there is an industry best practice to initially request funding sufficient to address the project’s planning and possibly requirements definition (Define) activities, to obtain answers to some of these questions, to re-validate the business case and then request the full project funding. This activity has been identified as obtaining “long lead funding.” Upon conclusion of the Plan or the Plan and Define, the project manager is better able to identify what costs will be incurred and when those costs will occur.

# Summary

Project life cycle models influence the planning and execution of projects. The Project Manager, working with his or her team, and using subject matter experts from the PMO and the Quality Organization should carefully consider the characteristics of the project about to be undertaken and select the most appropriate model for the project. All Project Team Members, including the Business Sponsors, need to understand the model before developing and approving the project schedule and estimates. Also, other project Stakeholders need to understand their role and responsibilities given that model.