

Prestwood Software Development Process™

Overview

PSDP Version 3.0 R1 – July 2002



PSDP is the easy-to-use, flexible, and scalable software development process.



Patent

Prestwood standards are prospective and advisory only. Prospective users are responsible for protecting themselves against liability for infringement of patents.

License to use this standard

Prestwood Software, the owner of the copyright of Prestwood Software Development Process™, hereby grant you a fully-paid up, non-exclusive, non-transferable, perpetual, worldwide license (without the right to sublicense), to create and distribute software and special purpose standards which are based upon this standard, and to use, copy, and distribute Prestwood Software Development Process.

Any unauthorized use of this standard may violate copyright laws, trademark laws, and communications regulations and statutes.

Notices

The information contained in this document is subject to change without notice.

The material in this document details a Prestwood standard. This document does not represent a commitment to implement any portion.

Disclaimer of Warranty

WHILE THE INFORMATION IN THIS PUBLICATION IS BELIEVED TO BE ACCURATE, this standard is provided "AS IS" and may contain errors or misprints.

The PSDP Documents

The PSDP standard is comprised of the following documents:

- PSDP Overview
- PSDP Project Management
- PSDP Analysis
- PSDP Development
- PSDP Quality Assurance
- PSDP Deployment
- PSDP Glossary

Additional supplementary material including additional documents, presentations, and the PSDP Templates are available at www.prestwood.com.

Credits

PSDP is a public standard and was first published as PSDM to the Internet in August of 1995. Although the controlling author of PSDP is Mike Prestwood, many have contributed to the definitions, concepts, philosophies, and content of PSDP. PSDP comes from a mixture of direct hands-on developing and managing of software development projects as well as Mike's interpretation of many contributors over the years. Those that deserve mention in this version of PSDP included:

Brian Prestwood
Carol Oberhaus
Dan Fought
Kim Berry
Randy Spitz
Scott Wehrly

Word Usage: Lessons, Suggestions, and Standards

Throughout PSDP terms like "Lesson", "Suggestion", and "Standard" are used carefully. A standard or rule must be followed in order to be compliant. A suggestion or guideline is optional. For example, a developer should use a suggested method for completing a task only when the developer doesn't already know a method that is more comfortable. Suggestions and guidelines are intended for less experienced developers. A lesson describes how to do a task. In general lessons do NOT belong as part of a standard. Lessons are included in this standard in order to educate or to exemplify. In general, a lesson deals with a specific part of a specific tool or method that is considered important enough and tricky enough to include.



Table of Contents

Section 1 - Introduction	1	Section 3 - Software Projects	7
1.1 About PSDP	1	3.1 Enterprise Modeling	7
1.2 What is process?.....	1	3.2 Project Feasibility.....	8
1.3 Scope of PSDP	1	3.3 Full Life-Cycle Software Development	8
1.4 PSDP Software Requirements	2	3.4 Documentation.....	9
1.5 The PSDP acronym:.....	2	Section 4 - The PSDP Phases	10
Section 2 - The PSDP Philosophy.....	3	4.1 Inception Phase.....	11
2.1 Minimalist Philosophy.....	3	4.2 Requirements Phase.....	11
2.2 Joint Responsibility.....	3	4.3 General Design Phase.....	11
2.3 Establish Success	4	4.4 Detail Design Phase.....	12
2.4 The Tough Stuff First.....	4	4.5 Initial Coding Phase.....	12
2.5 Natural Discovery.....	5	4.6 Testing & Rework Phase	12
2.6 Advanced Project Tracking	5	4.7 User Acceptance Phase.....	12
2.7 Customer Involvement with Iterative Methods	6	4.8 Deployment Phase.....	12
		Section 5 - Checkpoints and Compliance.....	13
		Section 6 - Rolling Estimates Example	15
		Section 7 - Where to go from Here	16



Section 1 - Introduction

1.1 About PSDP

Prestwood Software Development Process™ (PSDP) is a proven process that is scalability and flexibility geared toward managing the development of software systems, applications, components, web sites, and web applications. PSDP is an iteration-centric full life-cycle software development process. Iteration, as defined in PSDP, is a deployed version of the software to the end users. A project usually contains one iteration of the software numbered 1.0, 1.1, 2.0a, etc. Although PSDP is iteration-centric, it does offer mechanisms for enterprise modeling, multiple iterations, and the maintenance that occurs between iterations. PSDP can be used on projects using both outsourced and internal resources.

1.2 What is process?

In order to answer this question, PSDP defines standard, method, methodology, and then finally process (also known as a process pattern):

Standard	A best practice that is widely recognized or employed because of its excellence.
Method	A regular and systematic means or manner of procedure used to complete a project task or create a software artifact.
Methodology	A set of methods. For example, the Unified Modeling Language (UML) is a methodology that groups several methods together to document object oriented software including techniques for documenting requirements, class hierarchy, class interaction, etc.
Process / Process Pattern	A series of tasks performed in the making of software.

1.3 Scope of PSDP

Because of individuality, team dynamics, and company culture, a good software development process does not mandate every step. Instead, a good software development process clearly identifies a minimum set of required steps and provides suggestions (guidelines) and lessons for the rest of the steps.

Specifically, PSDP includes:

- Process Philosophy
- Process Term Definitions
- Process Concept Explanations
- Detailed Project Management Concepts and Steps
- Documentation Guidelines
- Development Guidelines
- Reviews and Testing Guidelines
- Deployment, Operations, and Maintenance Guidelines



1.4 PSDP Software Requirements

PSDP is an open standard and therefore does not have any software requirements. PSDP does include software suggestions and guidelines.

1.4.1 The PSDP Templates

Prestwood Software is providing the PSDP templates as a separate package for a nominal fee. These templates can be used along with PSDP to track and document a software project. Although PSDP does not require the use of the PSDP templates, the PSDP templates give you a framework that you can tailor to your specific needs. Visit www.prestwood.com for pricing and availability. For clarity, several of the templates are provided in PDF format as part of the downloadable version of PSDP.

1.5 The PSDP acronym:

Prestwood	Prestwood Software is the controlling organization of the PSDP standard. To suggest additions and changes, go to http://www.prestwood.com/standards/psdp .
Software	PSDP addresses all types of software development projects including software systems, software applications, components, web sites, and web applications.
Development	The PSDP standard helps in managing the building of custom software and is not intended for managing the purchasing of commercial or pre-packaged software.
Process	A process is a formula or recipe for how to accomplish something. A good software development process brings together various elements to guide and educate the development team and customer. This includes usage guidelines as well as required and suggested tasks, methods, methodologies, best practices, and leading technology.



Section 2 - The PSDP Philosophy

This section documents the philosophies and fundamental tenets that underlie PSDP and represents what makes PSDP different from other software development processes.

2.1 Minimalist Philosophy

A minimalist approach to software development is moderate or conservative. PSDP takes a minimalist approach toward software development. A minimalist philosophy helps keep costs down and the project pace up.

2.1.1 Easy to Use

A software development process must be easy to use or it will not be used on all projects. PSDP is based on a minimalist philosophy that permeates throughout with the minimum necessary project tasks, documentation, and testing required for the target iteration size and desired quality of the software. The target iteration size is the number of hours a development team is targeting to reach the target readiness date. The desired quality relates to the risk factors involved, the number of reviews used during the process, and the testing effort. When documenting a software development process, it is common practice to try to include every known step in an effort to make sure nothing is left out. This approach leads to an impressive but mostly unusable process. PSDP is lean and practical.

2.1.2 Deliver Quickly and Frequently

Because companies and organizations constantly evolve their business model, it is imperative to deploy software as quickly as possible. The project team should carefully plan the current and next few iterations. Deciding on what features to implement when is an important attribute of software development. The sooner the core features are implemented, the sooner the customer can start to recover a return on investment (ROI). Refer to the section titled, "Target Iteration Size" in the PSDP Project Management document for more information.

2.1.3 Keep it Simple Silly (KISS)

Software development is inherently difficult; therefore always go for the simpler solution you know best. This includes (but isn't limited to) the simplest architecture that will work and the simplest database solution that will work. If the customer needs a traditional multi-user application with a local database, do not propose client server or n-tier solution just because it would be more fun to do it that way.

Learn New Tools Before you Start a Project

If you know a development tool well, stick with it. Resist switching to the latest trend during a project. The time to explore and learn a new toolset is before starting a project. The problem is developers want to use the latest and greatest in an effort to stay up with technology. This is a good thing, of course, so a balance is in order.

2.2 Joint Responsibility

With PSDP, project success depends on the entire project team. The project team is comprised of both the development team and members representing the customer (user participants) and the executive sponsor. Although the project manager is the only project team member that must completely understand PSDP, all project members have responsibilities in PSDP. For example, the defining of the project scope is the executive sponsor's responsibility. And, although it is up to the project manager to educate the user participants with regard to their role within PSDP, the project will benefit if the user participants take responsibility for understanding PSDP and their role within it. Nowhere is this more important than in the joint responsibility of defining requirements and testing.

Project success is often contingent on how much feedback and input the user participants supply to the development team and the correct defining of the scope. Failure to perform these responsibilities in a timely manner, can lead to the project suffering possibly even resulting in an out of control budget, implementation of improper functionality, or even a failed project.



2.2.1 User Participant Testing Obligation

User participants must validate functionality and can help test for defects. PSDP distinctly separates these two disparate testing tasks. The specific testing strategy and test scripts are documented in the Test Plan document usually initiated during the requirements phase and finalized prior to the Testing & Rework phase.

Alpha Testing (Validate Functionality)

An alpha release has partial functionality implemented. The user participants validate the software functionality during the initial coding phase (defects are ignored during this process).

Beta Testing (Find and Prioritize Defects)

A beta release has all functionality implemented, but might contain defects depending on the level of unit testing performed by the developer. The level of customer involvement during beta testing is up to the customer. The more beta testing the customer agrees to perform, the less testing the development team will need to do. The user participants test the software for defects and help prioritize the defects as high, medium, or low. If missing functionality is discovered, it is prioritized and included in the appropriate future iteration.

Release Candidate Review (Final Review for High Priority Defects)

A release candidate has all functionality implemented, no known high priority defects, and the software has gone through a thorough level of testing usually including both running the test scripts (or equivalent) and regression testing (rechecking previous defects). The user participants review the software for high priority defects only.

2.3 Establish Success

For a project to be successful, success must be defined prior to starting the project. Because success varies from company to company and from project to project, PSDP establishes only general guidelines for success. It's up to the project team to establish success for a specific project. Establishing the initial critical success factors is part of the inception phase and is finalized during the requirements phase.

Success in the software development industry is often described as follows,

“Success is delivering software on time, within budget, to a satisfied user.”

This phrase is catchy and the project team should strive to reach its underlying philosophy. However, it is more practical to define success more precisely. PSDP defines both process success and project success. Process success is defined as part of PSDP and the customer defines project success in terms of critical success factors. The critical success factors are established in the inception phase and updated in the requirements specification. Project success can be defined in terms of budget, specific features, or both.

2.4 The Tough Stuff First

One of the reasons why projects end up taking longer than estimated and therefore costing more is because the tough stuff is left to the end or at least not accomplished first. Sometimes the development team takes too much time on the project “fine tuning” unimportant or at least less important features. After the customer has prioritized the desired features, the development team should work on the highest priority items first focusing on the features that are more technically difficult for the team to implement.

2.4.1 Proof of Concept Method

A proof of concept is a test of technology used to establish the truth. A proof of concept allows the developer to isolate a technology problem for testing. PSDP uses them at key points of the software development process. Proofs of concepts are particularly important when working with new technology or extreme business rules. For example, if the software must communicate to LED signs through a particular piece of hardware over the Internet, PSDP recommends the development team perform a proof of concept for that aspect of the software as soon as possible.



2.5 Natural Discovery

PSDP allows for natural discovery. For example, during the inception phase, the focus of the development team is on gathering high-level requirements with the user participants. Frequently the development team discovers detailed requirements during the inception phase. PSDP suggests the development team capture any detailed requirements in a rough Requirements Specification for later use.

2.6 Advanced Project Tracking

A good process has mechanisms for easy and accurate project tracking. In the large, PSDP marks the start and end of phases clearly with a single event. For example, the inception phase (the first phase) starts when discussion of a software project starts and ends when some type of agreement is executed. In the small, PSDP tracks phase tasks with checkpoints. For example, with the inception phase, the PSDP checkpoints include completing the inception phase worksheet (or equivalent), establishing the initial to-do list, optionally creating a project proposal, and finally executing some type of agreement. These tracking mechanisms are compatible with sequential, overlapping, and iterative project management.

2.6.1 Scalable

PSDP contains mechanisms that allow the project team to scale each project whether the project is a 40-hour project involving one or two project team members, or a five-year project involving a dozen project team members. Specifically, PSDP offers three iteration paths each with their own set of minimum checkpoints. The iteration paths are informal, formal, and robust. The project team chooses an iteration path based on the target iteration size and the desired quality of the software. *For more information on scale, estimating software and iteration size, and establishing desired quality, refer to the PSDP Project Management document.*

2.6.2 Flexible

Because of the varying talent level of project teams and varying cultures of organizations, a software development process must be flexible. PSDP is goal oriented and therefore gives a certain amount of flexibility to the development team. This includes a certain amount of flexibility for the project manager in how he or she manages the process and to the developers in what methods (techniques) are used to complete the project tasks. The project stays on track because PSDP establishes a minimum set of checkpoints (project tasks) based on the target iteration size and desired quality the project team must meet.

Intuitive versus Complete Methods

A Method is a discreet technique for accomplishing a development task such as creating a database, creating the class hierarchy, documenting the flow of data through a system, and documenting requirements. With PSDP, the developer can use both intuitive and complete methods throughout the project.

Method Type	Brief Description
Intuitive Method	An intuitive method relies more on the talent and experience of the developer to complete the specific development task than on the method.
Complete Method	A complete method is a documented step-by-step technique or approach for completing a discreet development task. A complete method relies more on the method than it does on the experience of the developer.



2.7 Customer Involvement with Iterative Methods

PSDP ensures customer involvement by employing iterative methods. Iterative methods are methods that move software artifacts back and fourth from the development team to the appropriate user participants. This process repeats as many times as needed to both validate functionality and to ensure the development team is focusing on the correct priorities. For time and material projects, the following represent typical software artifacts built using iterative methods.

Phase	Description of Iterative Software Artifacts
Inception	Iterative documents include the initial project to-do list and for fixed bids the description of deliverables. If there are lots of unknowns (such as budget or technology based unknowns), sometimes a formal feasibility study is undertaken during the inception phase. This is also a good time to do formal enterprise modeling.
Requirements	The primary iterative document is the requirements specification. For fixed bids, the description of deliverables is the requirements.
General Design	The iterative documents for the general design phase include one or more general design specifications.
Detail Design	For the detail design phase, the iterative documents vary depending on the software type but can include a detail design specification, a web page flow chart, an entity relationship diagram (ERD), a class hierarchy diagram, etc.
Initial Coding	The primary iterative software artifacts are various alpha deliverables. The focus is on validating functionality and not finding defects.
Testing & Rework	The primary iterative software artifacts are various beta deliverables. A beta deliverable is a functionally complete version of the software delivered for the purpose of identifying and prioritizing defects. The focus is on finding and prioritizing defects. If missing or incorrect functionality is found during this phase, something horrible has occurred during the process and the project team will need to take corrective action (which can include a mini corrective iteration).
User Acceptance	The primary iterative software artifacts are various release candidate deliverables. A release candidate has no known high priority defects. The focus is on reviewing the software for the purpose of determining if the build is deployable (the intention is to fix only high priority defects from this point through deployment).
Deployment	Iterative software artifacts include deployment plans, post deployment report, etc.

2.7.1 Iterative Methods and Fixed Bids

For fixed bids, the description of deliverables must be complete and precise. The focus of the development team must stay focused on the deliverables in the description of deliverables (nothing more and nothing less). The need for iterative methods is left entirely up to the development team. In general, fixed bids discourage the use of iterative methods. *Refer to the section titled, "Rolling Estimates Example" in this document for an alternative to fixed bids.*



Section 3 - Software Projects

Managing the development of custom software requires the identification of the strategic needs of the customer. This includes modeling the enterprise, discovering potential software projects, and the scope of each software project. This section identifies and briefly discusses software project concepts.

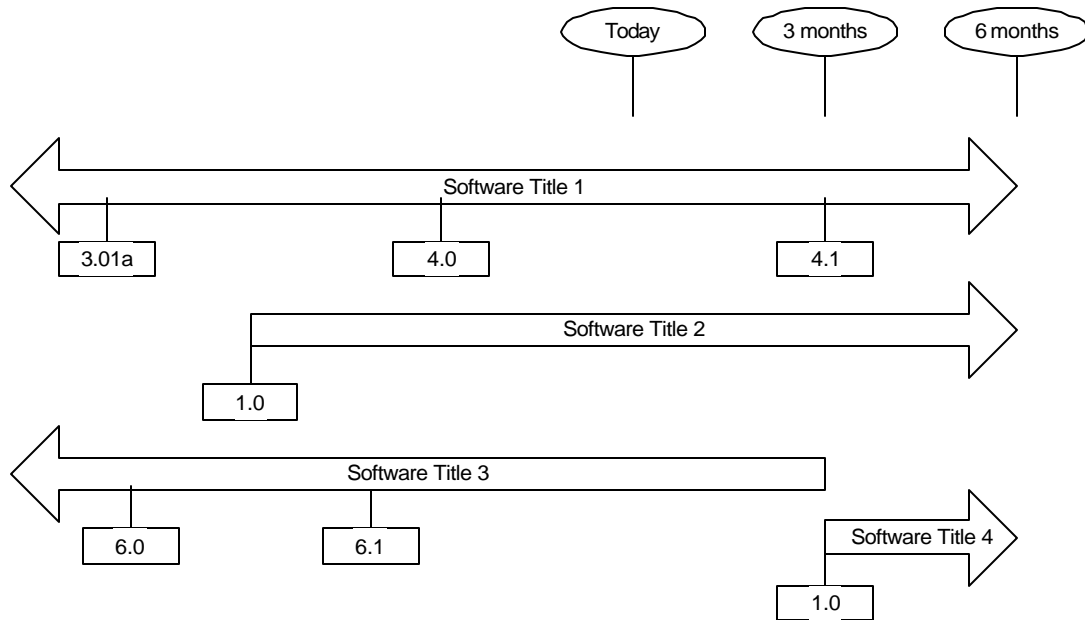
3.1 Enterprise Modeling

Custom software enterprise modeling is about discovering the custom software needs of a company or organization. At a minimum, model to achieve the following:

- Identify and Prioritize Custom Software Needs
- Identify any Opportunity for Reuse
- Consider Opportunity for Strategic Planning

PSDP includes both an intuitive and complete method for enterprise modeling. The intuitive approach relies heavily on the talent of the developer and all the tasks are done verbally with little to no documentation. The results are usually communicated verbally or in a short email. The complete approach relies more heavily on the process and the tasks are documented. The results are communicated with a formal enterprise model.

Graphic Depicting Enterprise Modeling



3.2 Project Feasibility

Sometimes a formal feasibility study is warranted to study whether a project is doable. This includes analyzing the budget, technology, scheduling, staffing, and other such aspects. A feasibility study addresses if the project can and should proceed. PSDP offers both an intuitive and complete method for assessing feasibility.

Feasibility Approach	Description
Intuitive Feasibility Method	The intuitive feasibility method relies more heavily on the talent and experience of the developer performing the feasibility study than on a step-by-step method. With the intuitive method, the feasibility results are often communicated verbally or with a short email.
Complete Feasibility Method	The complete feasibility method relies on a step-by-step process to put together a formal feasibility study document.

Feasibility of a project, whether you choose an intuitive or complete method, is always performed for every project, usually during the inception phase.

3.3 Full Life-Cycle Software Development

A full life-cycle software development process manages the various iterations of specific software from birth to death. Analysis of the software life-cycle will result in some type of iteration guidelines.

3.3.1 Projects & Iterations

As defined in PSDP, an iteration is a deployed version of the software to the end-users. The software is usually numbered 1.0, 1.1, 2.0a, etc. Although a project can consist of multiple iterations, a single iteration per project is more common. The more the development team tries to deliver in an iteration, the higher the risk. Because PSDP considers the use of divide and conquer a best practice, strive for smaller iterations. The first iteration should contain minimum core functionality required to start using the software.

3.3.2 Three Iteration Paths: Informal, Formal, and Robust

In order for an application development process to be scalable, it must address various target iteration sizes and desired qualities. For each phase within PSDP, there are three paths: informal, formal, and robust. Each path has a different set of minimum checkpoints. For example, a critical quality application estimated to take longer than one-man year would use the robust path for all phases.

When deciding on an informal, formal, or robust path for a particular phase of a project, a large aspect of what the project team is deciding is the level of documentation. This decision relates directly to the size and desired quality of the software. PSDP mandates a certain minimum level of documentation. So even when the development team decides to use an intuitive method for class design during the design phase, PSDP mandates the developer document the class hierarchy at a minimum. The Class Hierarchy checkpoint exists for both formal and robust iteration paths.

3.3.3 Multiple and Overlapping Iterations

Although PSDP focuses on a single iteration, there is nothing wrong with managing multiple and even overlapping iterations. This divide and conquer approach can lead to the deployment of core features to the end-users sooner and ultimately to success of the project.

3.3.4 Controlling Scope

Project scope is the range or area covered by the software being created. The scope of a software development project is explored during the inception phase and initially documented in the project proposal. Once scope is established, changes to the scope are referred to as scope creep.

For fixed bids, the project scope must be completely and precisely described in the description of deliverables. Changes in scope can only occur with an updated description of deliverables and a new agreement (or amendment).

For time and material projects, scope creep is natural and will occur. Having the ability in the process to handle scope creep is what is important. Scope creep is handled differently in PSDP depending on the billing relationship.



3.4 Documentation

During software development, information moves among the members of the project team. The managing of this information is really important. This section introduces the PSDP documentation. *For more information on PSDP documentation, refer to the PSDP document titled, "PSDP Sessions & Documentation".*

3.4.1 Software Documentation Overview

With PSDP you capture information in each phase moving, updating, and expanding as appropriate from one phase to another. The following introduces the documentation focus of each phase as well as what information is carried forward.

PSDP Phase	Documentation Notes
Inception Phase	Gather project parameters and general requirements and then determine feasibility and any enterprise modeling needs.
Requirements Phase	Carry forward updated project parameters and general requirements and then document the detailed requirements.
General Design	Carry forward updated project parameters and explore various general designs.
Detail Design	Carry forward updated project parameters and the chosen general design and then document the detailed design.

3.4.2 Common Word Method

This section gives you your first preview of the common word method used by PSDP to help manage information. Specifically, the common word method helps you easily determine where to store information.

Common Word	Brief Description
What	What the software should and must do is stored in the requirements specification.
Why	Why software is needed is contained in the business justification section of the project proposal.
How	How the software will address what is in the requirements specification is contained in the design documents. The primary design document used by PSDP is the design specification.
Where	Where the software will be deployed is stored in a UML deployment diagram or equivalent.
When	When software development is started is contained in the project to do list.

3.4.3 Initial, Updated, and Final Documentation

In PSDP, the three types of documents are initial, updated, and final. In PSDP, the iteration path indicates whether or not to create initial, updated, or finalized documents and at what stages in the development process.

Document Type	Description
Initial Document	This type of document is meant to capture the essence or bulk of the subject.
Updated Document	An updated initial document.
Final Document	This type of document is intended to serve as the final document (no updates will be needed). This doesn't mean that you can't update a final document, you can.



Section 4 - The PSDP Phases

This section introduces the PSDP phases. Refer to the PSDP document titled, "Project Management" for more information on the PSDP phases.

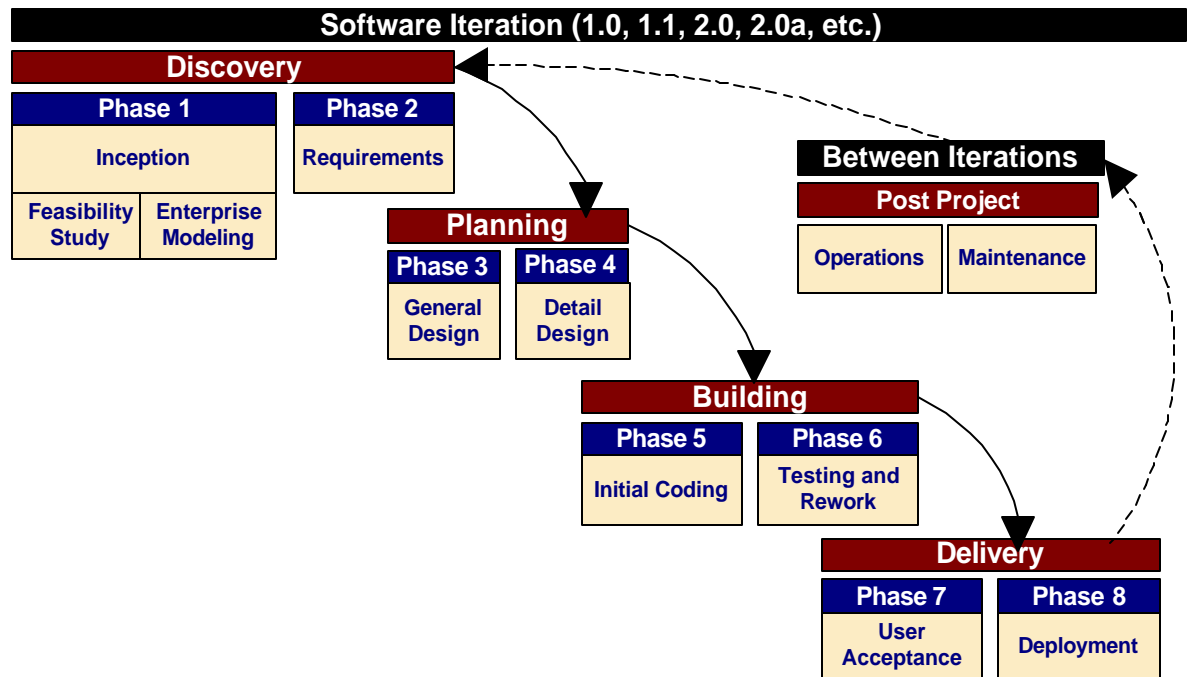
In the spirit of divide and conquer, each iteration proceeds in phases. In general, the phases are sequential, but many times overlap. For example, prior to completing the requirements phase, the development team will want to start the design phase. Starting the design phase as soon as possible both keeps the software project moving toward completion at a fast as reasonable pace, and draws out a few more requirements.

The following are the names of the phases that are used by PSDP followed by a very brief description.

1. **Inception** – Starting the project includes gathering project parameters such as feasibility, enterprise needs, general requirements, establishing the initial budget and billing relationship, etc.
2. **Requirements** – Documents what the software will do.
3. **General Design** – Explore various solutions for how the software will do the what gathered in the previous phase.
4. **Detail Design** – Architect the specific chosen solution.
5. **Initial Coding** – Build the software and validate functionality with alpha releases.
6. **Testing & Rework** – Test and fix the software with beta releases.
7. **User Acceptance** – Review the software for high priority defects. Is it ready to deploy?
8. **Deployment** – Deploy the software.

Graphic Depicting the Phases of an Iteration

The following graphic illustrates the iterative nature of software development and the phases of PSDP.



4.1 Inception Phase

The inception phase represents the initiating of the project. If performed properly, more projects than not should close during the inception phase, usually because of budget or scheduling issues.

The goals of the inception phase include gathering the project parameters and general requirements, creating a project proposal, and executing some type of agreement. The project parameters include such items as establishing dates such as the target start Date, identifying the project and software type, and desired quality. The general requirements include such items as establishing the general project definition, project scope definition, critical success factors, and scope defining specific features.

If the billing relationship will be fixed bid, then the description of deliverables is defined completely and precisely and a quote is given. If the billing relationship will be time and material, then execute an agreement and proceed with work. Work can proceed with an established working budget. For example, a duration based not to exceed clause may be added to the agreement. If the final price cannot be calculated, PSDP recommends establishing an initial working budget based on past similar projects and using the rolling estimates project management approach.

4.2 Requirements Phase

During the requirements phase, the majority of the project discovery is accomplished. The development team moves any general requirements gathered during the inception and feasibility phase into the requirements specification and expands them to sufficient detail to proceed into design. Although not part of the PSDP checkpoints, proof of concepts are encouraged.

The primary focus of the requirements phase is documenting, in writing, enough detailed requirements to proceed into the design phase. The goals of the requirements phase also include moving the project parameters and general requirements gathered during the inception phase into the requirements specification and updating and expanding as appropriate.

4.3 General Design Phase

The purpose of the general design phase is to explore and document possible software architectures and their respective pros and cons. During general design, the development team creates one or more general design proposals. Each proposal should have just enough detail in it to reasonably explore the pros and cons.

The goals of the design phase include exploring various technical solutions and choosing one general design proposal as the chosen general design.

General Design and Detail Design

Design in PSDP is handled in two distinct phases: general design and detail design phases. The general design phase documents one or more high-level designs of the software. Once a general design is chosen, the development team can proceed into detail design. In order to provide a customer with either a fixed bid or an accurate estimate, the development team must complete both the requirements phase and general design phase. Therefore, it is common to do the general design phase at the same time as the requirements phase.

Skipping General and Detail Design

Depending on the comfort level of the development team, the project manager may wish to skip either the general and/or detail design phases. Skipping either design phases doesn't mean the software isn't designed. It means the developer will design the software as they create it (during the initial coding phase). This technique works well for very small and small iterations with very talented developers.

The following documents the PSDP design options with regard to iteration paths.

	Informal	Formal	Robust
General Design	Opt	X	X
Detail Design		Opt	X



4.4 Detail Design Phase

The purpose of the detail design phase is to document the software design details in a design specification. If a recommended general design proposal exists, the information is moved from the general design proposal into the general design section of the design specification.

The primary focus of the detail design phase is documenting in writing enough design details required to proceed through the initial coding phase. The goals of the detail design phase also include moving the project parameters and general design into the design specification and updating and expanding as appropriate. For web sites, a web page flow chart is created. For GUI applications, a prototype is created. For database applications, database documentation is created.

4.5 Initial Coding Phase

During this phase, the development team constructs the software. The name of this phase is “initial coding” to stress the iterative nature with the testing & rework phase. The primary goal of the initial coding phase is to implement 100% of the functionality using the design deliverables.

The goals of the initial coding phase include constructing the software (implementing all functionality). During this phase, alpha deliverables are sent to the user participants to validate functionality. The focus is on validating that functionality meets the business need (not on finding defects).

4.6 Testing & Rework Phase

This phase includes defect testing only. The user participants should have validated the functionality, the feature set, during the initial coding phase with alpha deliverables. Testing types include system, integration, regression, and stress testing. During the testing & rework phase, the development team finds, prioritizes, and fixes defects.

Defect Priorities

Defects are prioritized as low, medium, or high priority. The priority indicates the need to fix the defect and not the effort to fix the defect. For example, although a typo is easy to fix, it will probably be a high priority defect because it must be fixed. Keep in mind that any defect can change priority at any time.

Defect Priority	Brief Description
Low	A low priority defect does not need to be fixed.
Medium	A medium priority defect may need to be fixed but is for one reason or another, considered optional. For Formal and Robust paths, medium and high priority defects must be fixed prior to deployment.
High	A high priority defect is a defect that must be fixed.

4.7 User Acceptance Phase

The purpose of the user acceptance phase is to identify any remaining high priority defects with the ultimate goal of the customer accepting the final product.

4.8 Deployment Phase

This phase is, of course, the final step in completing a project. For many projects, this can be as simple as sending setup files via email or on a CD to the customer. For some projects, however, the deployment phase could be rather complicated. For example, data migration from an old to a new system can mean additional tasks including a rollback plan.



Section 5 - Checkpoints and Compliance

A checkpoint is a review, deliverable, or the completion of a specific task. Within PSDP there are required and optional checkpoints. These checkpoints mark the progress of a project. The required checkpoints represent the minimum checkpoints the development team must complete in order for the project to be PSDP compliant. The number of required checkpoints is scalable with the size and quality of the project and is represented by the informal, formal, and robust paths.

5.1.1 Inception Phase Checkpoints

	Informal	Formal	Robust
Gather Project Parameters	X	X	X
Gather General Requirements	X	X	X
Establish Billing (fixed or T & M, price or working budget, etc.)	X	X	X
Project proposal	Opt	Opt	Opt
Execute Agreement	X	X	X

5.1.2 Requirements Phase Checkpoints

	Informal	Formal	Robust
Database Software: Initial Entity Relationship Diagram (ERD) (This initial or draft ERD is created during the requirements phase to draw out requirements.)		X	X
GUI Software: Initial Visual Flow Chart		X	X
Sunny Day Use Case Diagrams with Use Case Text		X	X
Final Requirements Specification	X		
Initial Requirements Specification		X	X
Updated Project To-Do List	X	X	X

5.1.3 General design phase minimum checkpoints

	Informal	Formal	Robust
General Design Session		Opt	X
Choose a recommended General Design Proposal		Opt	X
Deliver General Design Proposal(s)		Opt	X

5.1.4 Detail design phase minimum checkpoints

	Informal	Formal	Robust
Update Project To Do List	X	X	X
Database Software: ERD		X	X
Database Software: Data-Dictionary			X
GUI Software: Finalize Visual Flow Chart			X
GUI Software: Prototype (All apps with a GUI)	X	X	X
UML Class Hierarchy Diagram		X	X
Common UML Class Interaction Diagrams			X
Design Specification		X	X
Updated Requirements Specification			X
Final Requirements Specification		X	
Establish Baseline		X	X



5.1.5 Initial Coding Checkpoints

	Informal	Formal	Robust
To-Do List Updates		Weekly or Monthly or by Module Plus as requested	Weekly or by Module Plus as requested
Weekly Executive Statuses	X	X	X
Validate Functionality with Alpha deliverables	X	X	X
Interim Code Reviews (either weekly or by module are the recommended methods)			X
T & M: Review Software Against Requirements			X
Fixed Bids: Review Software functionality against description of deliverables	X	X	X
Final Code Review		X	X

5.1.6 Testing & rework phase minimum checkpoints

	Informal	Formal	Robust
Create Test Plan		X	X
Test for Defects	X	X	X
Prioritize Defects as Low, Medium, and High	X	X	X
Fix Medium Priority Defects		X	X
Fix High Priority Defects	X	X	X

5.1.7 User acceptance phase minimum checkpoints

	Informal	Formal	Robust
Deliver Release Candidate 1	X	X	X
Customer Review for High Priority Defects	X	X	X
Fix ALL High Priority Defects	X	X	X
Final Release (Contains no known high priority defects.)	X	X	X

5.1.8 Deployment phase minimum checkpoints.

	Informal	Formal	Robust
Deployment Plans*		X	X
Final Requirements Specification			X
Final Design Specification			X
Deploy Software	X	X	X
Train Users		X	X
Train Admin(s) on Operational Needs		X	X
Post Deployment Report		X	X

*The deployment plans include the deployment plan, rollback plan, training plan, and operational plan.



Section 6 - Rolling Estimates Example

The rolling estimates example discussed in this section offers more control over the software project budget with less risk for both the customer and development team. *For more information on the Rolling Estimates Project Management Approach, refer to the PSDP Project Management document.*

Rolling Estimates Project Management Example

Establishing an accurate estimate for a task, phase, or software project can be tricky. The larger it is, the more likely the estimate will be wrong. With the rolling estimates project management approach, both the customer and development team can proceed with the project with very little risk. With rolling estimates project management, the development team gives final estimates for the next phase only and working estimates for the rest. These estimates are used to either establish the next budget or to cancel the project. The amount of actual work performed for a task with a final estimate is expected to stay within the estimate. However, the amount of actual work performed for a task with a working estimate is expected to differ from the estimate.

The following describes applying the rolling estimates project management approach to a new software project.

Phase Ending	Description of Estimate(s)
Inception Phase	During the inception phase, the project team establishes the necessary factors required to start the project. This includes establishing the approach. Estimates: <ul style="list-style-type: none">• Final estimate for starting phase (usually the requirements phase).• Working estimates for the rest of the phases may also be given. These estimates will change.
Requirements Phase	Final estimate for the general design and detail design phases. Updated working estimates for the rest of the phases.
Detail Design	Final estimate for the initial coding phase. Updated working estimates for the rest of the phases may also be given.
Next Phase	Final estimate for the next phase or tasks. Updated working estimates for the rest of the phases may also be given.

Establishing a Working Budget

Depending on many factors, the project team works to establish an approved working budget based on current knowledge of the project and the desired project approach. The desired approach is an agreement between the development team and the executive sponsor on how to proceed. Although many project approaches are possible, there are two common approaches.

Project Approach	Description
Task Oriented	A task oriented approach calls for the development team to work on a given task or portion of the project such as enterprise modeling, feasibility, requirements, or design. When that task or tasks is completed or the budget is used up, the development team stops work until a new budget is approved.
Rolling Budget	The rolling budget approach calls for the development team to work on the project until completed. This approach better allows for overlapping phases, which can dramatically speed up the schedule. The project team can still establish a duration-based budget. If a duration-based budget is established, development stops when either the budget is used up or the project is completed.



Section 7 - Where to go from Here

The purpose of this document was to familiarize you with PSDP. If you're interested in learning more about PSDP you have plenty of free resources at your disposal including complete PSDP documentation, supplementary articles, online presentations, and an online message board. If your organization is interested in adopting PSDP as your process standard then the PSDP templates and project management certification are also available for a nominal fee. Prestwood Software also offers enterprise-level services including custom training, independent process verification, project outsourcing, and staff augmentation.

PSDP Online

The PSDP online resources at <http://www.prestwood.com/psdp> include:

- **All The PSDP Documents** – Download the complete standard.
- **Online Presentations** - These self-paced tutorials include spoken help from Mike Prestwood (the primary author of PSDP).
- **Online Presentations (no sound)** - Also available for download without audio are the same presentations. These presentations are for your use and customization with your staff and customers.
- **Supplementary Articles** – Supplementary articles include feasibility, OO, UML, DFD, etc.
- Online Community
- **PSDP Message Board** – Discuss PSDP and software development process.

PSDP Templates

Prestwood Software is providing the PSDP templates as a separate package starting at \$125. These templates can be used along with PSDP to track and document a software project. The PSDP templates give you a framework that you can tailor to your specific needs. For more information, go to <http://www.prestwood.com/standards/psdp/templates.html>.

Custom Training

Prestwood Software offers training in PSDP. If your company or organization is adopting PSDP and you need custom training, then contact Prestwood Software.

PSDP Project Manager Certification

If you're a project manager and you're interested in becoming a certified PSDP project manager, go to <http://www.prestwood.com/standards/psdp/certification.html>.

