



Development Phases

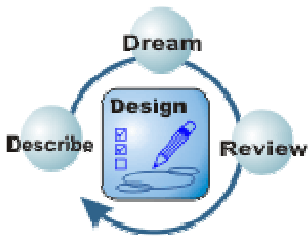
The development effort is composed of four general phases. Each of these phases is very iterative and blur together near the boundaries. They are listed in roughly the order of implementation, but sometimes they can overlap or be performed somewhat in parallel. Each element of the project will go through these phases, but not always at the same rate. One element can be prototyped while another is being developed.

The diagrams depict the main flow of the category as well as the sub-elements of effort.

The purpose of describing these categories is to help planning and collaboration. This creates a common understanding and frame of reference for the development effort.

Design

Create the functional and technical documentation that will be the blue print for the system.



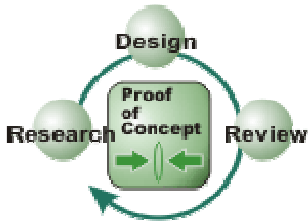
Design Models: Design Models describe and depict the project's features and intent – the what and why. The goal is to define the high order elements, capabilities, and interactions. This will be used during the functional specification. Another goal is to start making decisions about what functionality will be included in Version 1 and what will need to be included in a later version. Design Models use textual descriptions, examples and user interface sketches.

Functional Specification: Determine what the system will do – its functions. This is like the architectural drawings for a building – determine its use and structure but not the individual building parts. It uses the Design Models to determine the functions of the system.

Guide Lines: Create the guide lines for how the system will be developed and tested. This includes Development Principles, Coding Guidelines, Testing Guidelines, and Technology Manifest.

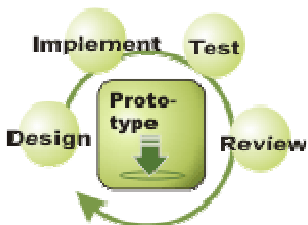
Technical: Determine how the system will be constructed (the technology and methodology) to meet the functional specification. This is like a blue print for a building – it is derived from the architectural drawings, but tells specifically how the building will be built.

Proof-of-Concept



Use the technical specification and create proof-of-concept elements of the software. These elements may not be built as interacting elements of the project – the point is to study and work on the mechanisms and concepts of the project. At the end of this phase there should be a base of understanding and some code that will be used in the final system. In essence this is like sketching a scene before painting it, or painting a scene as small concept works before choosing and beginning to paint the final work. At this point most of the proof-of-concept code will need re-factoring.

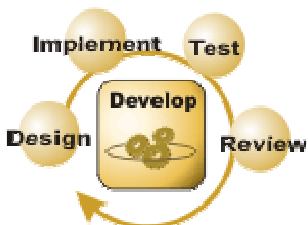
Prototypes



Alter the technical specification as needed based on the proof-of-concept phase. Use the technical specification and proof-of-concept understanding to begin building the prototype elements of the project. Alter the technical and functional specifications if required. At the end of this phase there should be a deep understanding, code patterns, and some structure build the elements of the project. Using the painting analogy again, this phase is like preparing the canvas, chalking some outlines and even applying the initial background washes.

Late phase prototypes should undergo extensive internal testing and some external alpha testing.

Develop



Use the prototypes and specifications to build the elements of the system. At the end of this phase the bulk of the coding should be done and the elements should be working – but not necessarily bug free.

Mid to late in this phase there should be extensive internal testing and external beta testing. At the end of this phase there should be some early adopter testing.