SOFTWARE DEVELOPMENTMETHODOLOGIES



Impetus Technologies, Inc.

5300 Stevens Creek Boulevard,

Suite 450, San Jose,

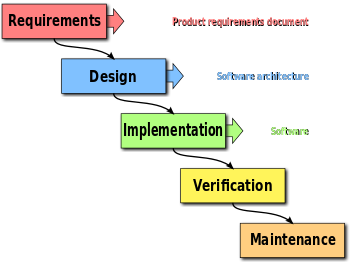
CA. 95129, USA

[www.impetus.com](http://www.impetus.com)

# INTRODUCTION: A software development methodology or system development methodology in software engineering is a framework that is used to structure, plan, and control the process of developing an information system. Common methodologies include waterfall, prototyping, iterative and incremental development, spiral development, rapid application development, and extreme programming. A methodology can also include aspects of the development environment (i.e. IDEs), model-based development, computer aided software development, and the utilization of particular frameworks (i.e. programming libraries or other tools).

1. APPROACHES: There are several approaches which any Enterprise application can follow for the development cycle. They can be classifies as given below:
2. Waterfall Model: The waterfall model is sequential approach in which development is seen as flowing steadily downwards (like a waterfall) through the phases of requirements analysis, design, implementation, testing (validation), integration, and maintenance. The basic principles are:

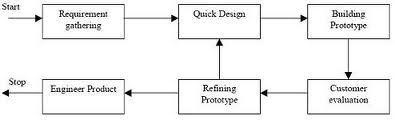
* Project is divided into sequential phases, with some overlap and splash back acceptable between phases.
* Emphasis is on planning, time schedules, target dates, budgets and implementation of an entire system at one time.
* Tight control is maintained over the life of the project via extensive written documentation, formal reviews, and approval/signoff by the user and information technology management occurring at the end of most phases before beginning the next phase.
* Requirements are frozen in initial stage and any new changes is not accepted with ease in this approach.

[](http://en.wikipedia.org/wiki/File:Waterfall_model.svg)

**Water Fall Model**

1. Prototyping: The development approaches involves creation of software prototypes. The basic principles are:

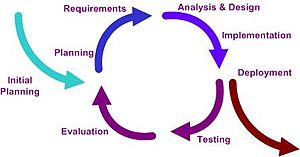
* Not a standalone, complete development methodology, but rather an approach to handle selected parts of a larger, more traditional development methodology.
* User is involved throughout the development process, which increases the likelihood of user acceptance of the final implementation.
* Small-scale mock-ups of the system are developed following an iterative modification process until the prototype evolves to meet the users’ requirements.
* While most prototypes are developed with the expectation that they will be discarded, it is possible in some cases to evolve from prototype to working system.

[](https://www.google.co.in/search?tbm=isch&q=prototype+model+sdlc&revid=1161298119)

**Prototyping Model**

1. Incremental and Iterative: Various methods are acceptable for combining linear and iterative systems development methodologies, with the primary objective of each being to reduce inherent project risk by breaking a project into smaller segments and providing more ease-of-change during the development process. The basic principles are:

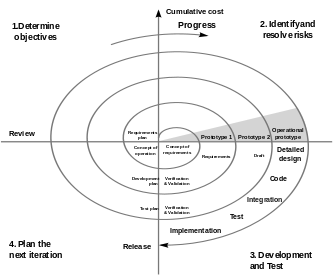
* A series of mini-Waterfalls are performed, where all phases of the Waterfall are completed for a small part of a system, before proceeding to the next increment, or.
* Overall requirements are defined before proceeding to evolutionary, mini-Waterfall development of individual increments of a system, or.
* The initial software concept, requirements analysis, and design of architecture and system core are defined via Waterfall, followed by iterative Prototyping, which culminates in installing the final prototype, a working system.

[](http://en.wikipedia.org/wiki/File:Iterative_development_model_V2.jpg)

**Incremental and Iterative Model**

1. Spiral Development: It follows a top-down, bottom-up approach by combining elements of designing and prototyping. The basic principles are:

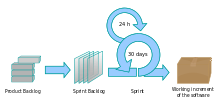
* Project is broken into smaller segments and providing more ease of change during development process.
* Each time the complete cycle is followed.

[](http://en.wikipedia.org/wiki/File:Spiral_model_(Boehm,_1988).svg)

1. Agile: Agile software development is a group of software development methods based on iterative and incremental development, in which requirements and solutions evolve through collaboration between self-organizing, cross-functional teams. It promotes adaptive planning, evolutionary development and delivery, a time-boxed iterative approach, and encourages rapid and flexible response to change. It is a conceptual framework that promotes foreseen tight iterations throughout the development cycle.

The basic principles are:

* Customer satisfaction by rapid delivery of useful software.
* Welcome changing requirements, even late in development.
* Working software is delivered frequently (weeks rather than months).
* Close, daily cooperation between business people and developers.
* Projects are built around motivated individuals, who should be trusted.
* Face-to-face conversation is the best form of communication (co-location).
* Working software is the principal measure of progress.
* Sustainable development, able to maintain a constant pace.
* Continuous attention to technical excellence and good design.
* Simplicity—the art of maximizing the amount of work not done—is essential.
* Self-organizing teams.
* Regular adaptation to changing circumstances.

[](http://en.wikipedia.org/wiki/File:Scrum_process.svg)

**Agile- Scrum Model**

1. WHY AGILE- SCRUM: We have chosen Agile – Scrum methodology for our Engineering Excellence tasks. Reasons are cited below:
2. Requirements may change even at the last moment so the approach should be flexible to accept such changes.
3. Close and daily cooperation between team members is involved.
4. The task can be split into User stories which can form the Product Backlog.
5. Agile is intended for mature developers, and engineering excellence should be used as a platform to inculcate software development maturity in Impros.
6. Different developers will work on different module.
7. The methodology will help us in distributing tasks to team members module-wise.
8. The methodology will help in developing self-managed attitude in software development process.
9. REFERENCES:

<http://smallbusiness.chron.com/selfmanaged-team-18236.html>

<http://en.wikipedia.org/wiki/Software_development_methodology>

<http://spin.atomicobject.com/2013/09/27/design-agile-waterfall/>

<http://www.itinfo.am/eng/software-development-methodologies/>