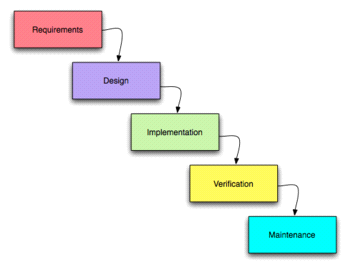
**SDLC Models**

* **Water fall model**
* **Spiral model**
* **Incremental model**
* **Prototype model**
* **Rapid development model**
* **V model**
* **Agile model**
* **Water-Fall Model**
* In water-fall model the system or project is modeled in sequential manner. i.e. The SDLC phases will be accomplished in linear fashion (one after the another)
* One activity can’t be started until the previous activity is completed.

**For ex:**

* Testing can't be started until all previous phases are finished.
* Development activities carried out sequentially
* Review and approval of each phase outputs
* If any defect found, go back to the originating phase and start traversing sequentially all over again
* This model is also termed as **Linear-Sequential Model**



**SDLC Workflow**

* Suitable for projects where:–
* Requirements are clearly defined
* Small and medium term duration
* Stable technology
* Familiarity with the domain and development environments

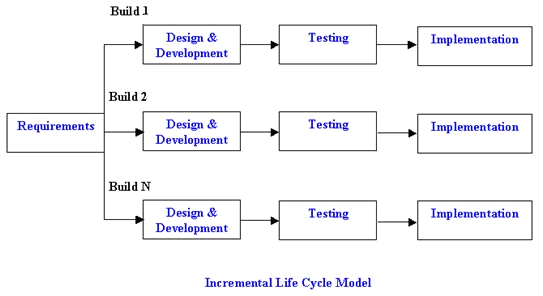
**Advantages:-**

* Projects under control
* Pre-defined outputs at every phase
* Tracking changes is easy
* Early identification of slippages, if any

**Disadvantages:-**

* In real life, customer requirements do change
* Customer appraisal of completed work - not feasible always
* Phases cannot run concurrently

**Incremental Model**



* Multiple development cycles take place here, making the life cycle a “multi-waterfall” cycle.  Cycles are divided up into smaller, more easily managed iterations.
* Each iteration passes through the requirements, design, implementation and testing phases.
* Subsequent iterations build on the initial software feedback from customer
  + - 1. [Build#2 developed based on Build#1 feedback]
* **Advantages:**
* Generates working software quickly and early during the software life cycle.
* More flexible – less costly to change scope and requirements.
* Easier to test and debug during a smaller iteration.
* Easier to manage risk because risky pieces are identified and handled during its iteration.
* Separate testing team required for testing because testing conduct

Very early in development life cycle.

Disadvantages:🡪

* Separate risk analysis team required to manage risks, Because

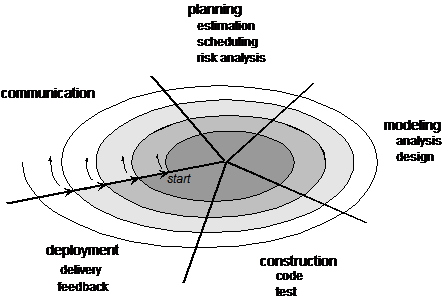
Software is developing in bits and peaces

* Separate Operational team required to manage number of

Of iterations.

**Spiral Model**

* The spiral model is similar to the incremental model, with more emphases placed on risk analysis.
* The spiral model has four phases: Planning, Risk Analysis, Engineering and Customer Evaluation.
* A software project repeatedly passes through these phases in iterations (called Spirals in this model).
* The spiral model used in combines the features of the waterfall model and the prototyping mode.
* Each Spiral consists of a deliverable product
* Feedback of each spiral is incorporated in the next spiral
* Customer can start using the system after every spiral
* Each Spiral consists of a waterfall model



**Spiral Model –**

* **Advantages:**
* Good for mission-critical projects.
* Software is produced early in the software life cycle
* **Disadvantages:**
* Can be a costly model to use.
* Highly large skilled people in the area of planning, risk analysis and mitigation, development, customer relation etc. are required.
* Risk analysis requires highly specific expertise.
* Project’s success is highly dependent on the risk analysis phase.
* Doesn’t work well for smaller projects.

**Prototype Model**

* A prototype model is a representation of a real life situation, which can be evaluated by the user.
* A prototype is developed based on the initial understanding of the customer requirements.
* A visible working prototype helps customer to define the requirements.

**Advantages:**

* Can be used when customer is not sure about what he wants.
* Faster way of finalizing the requirements.
* Useful for new technologies and domains.

**Disadvantages:**

* Since its being built out of concept, most of the models presented in the early stage are not complete.
* Since the prototype changes from time to time, it?s a nightmare to create a document for this software.
* There are many things that are removed, changed and added in a single update of the prototype and documenting each of them has been proven difficult.

**Rapid Development Model**

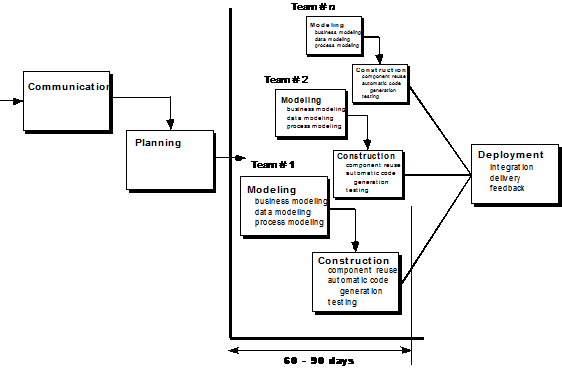
* RAD is a high speed adaptation of linear sequential model by using component based construction
* RAD enables creation of fully functional software within a very short time

**Advantages :**

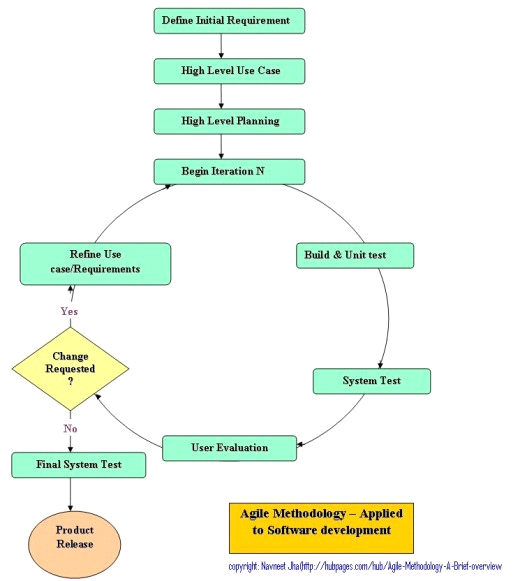
* Very short cycle time
* Use of working components lead to minimum defects

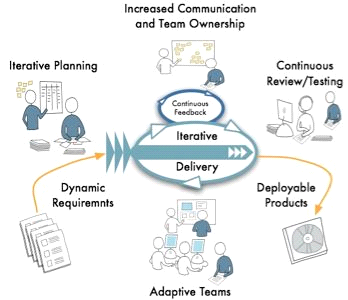
**Disadvantages :**

* Not useful for systems that can NOT be properly modularized

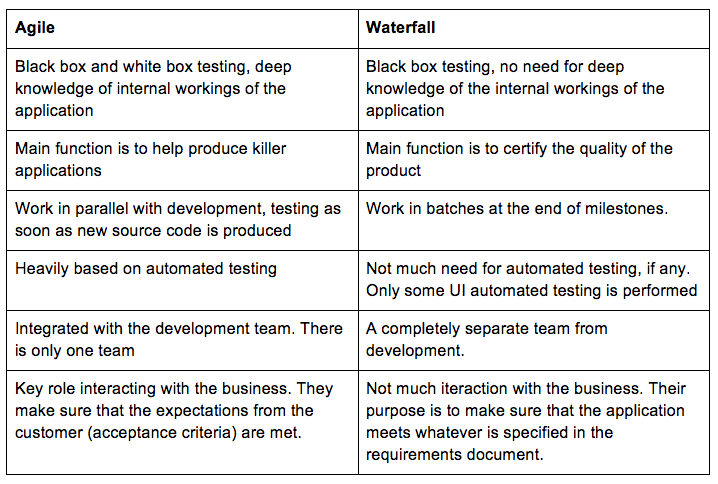


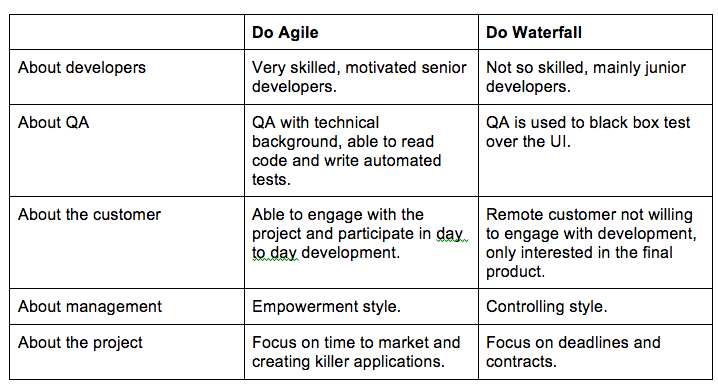
**Agile Model :**





* Agile is most important practical characteristic is that it is based on iterations. An iteration in Agile is a small period of time, usually from 1 week to 4 weeks, where some functionality for the application is built from end to end.
* The agile models, as the title suggest, focuses on 'agility' and 'adaptability' in development.
* Instead of one time consuming and rigid development schedule, agile models involve multiple iterative development schedules that seek to improve the output with every iteration
* Each iteration goes through the all the steps of design, coding and testing. The design is not set in stone and is kept open to last minute changes due to iterative implementation.
* Correcting the various bugs that arise over time.



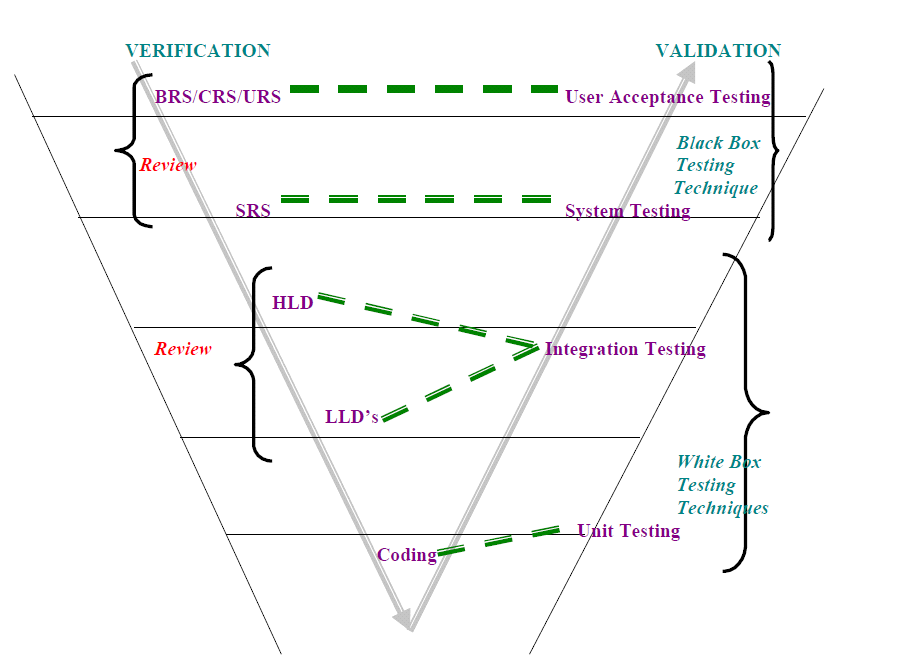


**V Model**

* The V-model gives equal weight to coding and testing in a software development process.
* The V-Model is a Software life cycle model.
* It explains, how you go from high level concept to released product.
* At the top of the left V line, is the High level concepts.
* At the bottom of the V is the coding phase.
* At the top right V line is the released product.
* The reason for calling it a V is because most of the phases on the left has a corresponding phase of activity on the right.
* From v model onwards organization started recruiting separate software

Testing team

* For more details about V Model, visit:



Difference s between verification and validation testing

|  |  |
| --- | --- |
| Verification Testing | Validation Testing |
| 1. Also known as static testing | 1. Also known as dynamic testing |
| 1. It is a document level testing | 2. It is a software level testing |
| 3. During this testing tester only  concentrate on reviews and walkthroughs around high level documents like [BRS,SRS,HLD,LLD] | 3. During this test testing  Team operate real software and validate expected behavior |
| 4. It is defect preventive approach | 4. It is defect detective approach |
| 5. Are we developing right system are not | 5. Developed software meeting with client expectations are not |