SOFTWEARE DEVELOPMENT LIFE CYCLE(SDLC)

SDLC provides a well-structured flow of phases that help an organization to quickly produce high-quality software which is well-tested and ready for production use. The SDLC involves six phases as explained in the introduction. Popular SDLC models include the waterfall model, iterative model, spiral model, v – model, Rad model and Big Bang model.

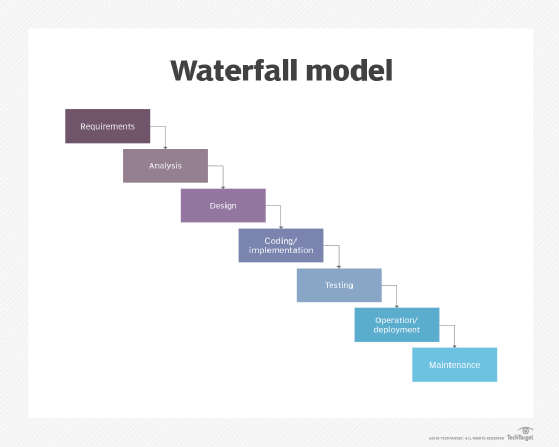
# WATERFALL MODEL

The waterfall model is a linear, sequential approach to the [software development lifecycle](https://www.techtarget.com/searchsoftwarequality/definition/systems-development-life-cycle) (SDLC) that is popular in software engineering and product development.

The waterfall model uses a logical progression of SDLC steps for a project, similar to the direction water flows over the edge of a cliff. It sets distinct endpoints or goals for each phase of development. Those endpoints or goals can't be revisited after their completion.

Who uses the waterfall model?

Project teams and Project managers use the waterfall model to achieve goals based on the needs of their business. The model is used in many different project management contexts, such as in construction, manufacturing, IT and software development.



**Requirements.** Potential requirements, deadlines and guidelines for the project are analyzed and placed into a formal requirements document, also called a [*functional specification*](https://www.techtarget.com/searchsoftwarequality/definition/functional-specification). This stage of development defines and plans the project without mentioning specific processes.

**Analysis.** The system specifications are analyzed to generate product models and [business logic](https://www.techtarget.com/whatis/definition/business-logic) to guide production. This is also when financial and technical resources are audited for feasibility.

**Design.** A design specification document is created to outline technical design requirements, such as the programming language, [hardware](https://www.techtarget.com/searchnetworking/definition/hardware), data sources, architecture and services.

**Coding and implementation.** The [source code](https://www.techtarget.com/searchapparchitecture/definition/source-code) is developed using the models, logic and requirement specifications designated in the prior phases. Typically, the system is coded in smaller components, or units, before being put together.

**Testing.** This is when [quality assurance](https://www.techtarget.com/searchsoftwarequality/definition/quality-assurance), [unit](https://www.techtarget.com/searchsoftwarequality/definition/unit-testing), [system](https://www.techtarget.com/searchsoftwarequality/definition/system-testing) and [beta](https://www.techtarget.com/whatis/definition/beta-test) tests identify issues that must be resolved. This may cause a forced repeat of the coding stage for [debugging](https://www.techtarget.com/searchsoftwarequality/definition/debugging). If the system passes integration and testing, the waterfall continues forward.

**Operation and deployment.** The product or application is deemed fully functional and is deployed to a live environment.

**Maintenance.** Corrective, adaptive and perfective maintenance is carried out indefinitely to improve, update and enhance the product and its functionality. This could include releasing [patch](https://www.techtarget.com/searchenterprisedesktop/definition/patch) updates and new versions.

**Advantages of the waterfall model**

Today, Agile methodology is often [used in place](https://www.pmi.org/learning/library/agile-versus-waterfall-approach-erp-project-6300) of the waterfall model. However, there are advantages to the waterfall approach, such as the following:

enables large or changing teams to move toward a common goal that's been defined in the requirements stage;

forces structured, disciplined organization;

simplifies understanding, following and arranging tasks;

facilitates departmentalization and managerial control based on the schedule or deadlines;

reinforces [good coding habits](https://www.techtarget.com/searchsoftwarequality/feature/Learn-5-defensive-programming-techniques-from-experts) to define before implementing design and then code;

enables early system design and specification changes to be easily done; and

clearly defines milestones and deadlines.

**Disadvantages of the waterfall model**

Disadvantages of the waterfall model typically center around the risk associated with a lack of revision and flexibility. Specific issues include the following:

Design isn't adaptive; when a flaw is found, the entire process often needs to start over.

Method doesn't incorporate midprocess user or client feedback, and makes changes based on results.

Waterfall model delays testing until the end of the development lifecycle.

It doesn't consider error correction.

The methodology doesn't handle requests for changes, [scope](https://www.techtarget.com/searchcio/definition/project-scope) adjustments and updates well.

Waterfall doesn't let processes overlap for simultaneous work on different phases, reducing overall efficiency.

No working product is available until the later stages of the project lifecycle.

Waterfall isn't ideal for complex, high-risk ongoing projects.

# ITERATIVE MODEL

The iterative process model is the implementation of the software development life cycle in which the initial development is started based on the initial requirements and more features are added to the base software product with the ongoing iterations until the final system is created.

In this model, the total software development is divided into iterations and each iteration has design, development, testing and review. The remaining phases, planning and requirement analysis, deployment and maintenance are one time and doesn't involve in interations.



**Advantages of Iterative Model**

* This model produces a working software much quickly and early during the SDLC.
* This model is very flexible. As new functionality can be added to it at any time of development.
* This model is considerably cheap as it is less costly to change requirements as compared to the other process models.
* The end-user or the stakeholders can give their feedback quickly, which can then be implemented into the system.
* The errors and bugs in the system can be identified early.
* Takes smaller development teams as compared to other process models.

**Disadvantages of Iterative Model**

* Problems pertaining to the system architecture can come up because all the requirements are not gathered upfront.
* It is not a good choice for small projects.
* More resource-intensive than waterfall model.
* Risk analysis requires highly qualified specialists to check the risks in our system.
* The whole process is difficult to manage.

# SPIRAL MODEL

The spiral model is a systems development lifecycle (SDLC) method used for risk management that combines the iterative development process model with elements of the Waterfall model. The spiral model is used by software engineers and is favored for large, expensive and complicated projects.

**Risk handling in spiral model**

The spiral model analyzes all proposed solutions and identifies, analyzes and addresses all potential risks. Following this, methods such as prototyping, simulation, benchmark testing, analytical models, and user research are used to develop the lowest-risk, most cost-effective strategy.

**Spiral model phases**

The different phases of spiral model are-

**1. Planning**

This phase begins by gathering  business requirements into a baseline spiral. In the subsequent spiral as the product matures, all system, subsystem  and unit requirements are identified at this stage.

This phase also includes understanding  system requirements through ongoing communication between the customer and  system analysts. At the end of the spiral, the product will be deployed in the identified market. This includes iteration cost, schedule, and resource estimates. This includes understanding  system requirements for ongoing communication between  system analysts and customers.

**2. Risk Analysis**

After the “plan” phase, the team prepares for the “risk” phase. The “risk” phase is designed to take into account the variability in the rate at which a given product might fail. It is designed to account for the uncertainty in the rate at which a given product might fail. During the “risk” phase, the team evaluates various aspects of the current state of the product, such as the state of its code, the state of its design, and the state of its prototype. The team then makes adjustments to the current state of the product based on the changes made in the “plan” phase, and then follows up with a “sales” phase to collect customer feedback.

Once  risks are identified, risk mitigation strategies are planned and completed.

Briefly, risk analysis involves identifying, estimating and monitoring  technical feasibility and management risks such as: schedule slippage and cost overrun. After testing the build, at the end of the first iteration, customers rate the software and provide feedback.

**3. Product development**

In the next quadrant, prototypes are built and tested. This step includes architectural design, module design, physical product design and  final design. Convert the proposals made in the first two quadrants  into usable software.

This phase also includes the actual implementation of features in a project which are verified by performing testing.

4**. Next phase planning**

In this phase ,the software is evaluated by the customer and feedback is given. The team prepares for the next phase of the planning process. The next phase of the planning process is known as the “spiral” phase. During the “spiral” phase, the team determines the order of events in the current state of the product and then follows these events up with a “revision” phase to “Revise” the current state of the product so that it is ready for production. The “revision” phase is also called the “reproduction” phase, and it is one of the most important aspects of the planning process.



The advantages of the Spiral SDLC Model are as follows −

* Changing requirements can be accommodated.
* Allows extensive use of prototypes.
* Requirements can be captured more accurately.
* Users see the system early.
* Development can be divided into smaller parts and the risky parts can be developed earlier which helps in better risk management.

The disadvantages of the Spiral SDLC Model are as follows −

* Management is more complex.
* End of the project may not be known early.
* Not suitable for small or low risk projects and could be expensive for small projects.
* Process is complex
* Spiral may go on indefinitely.
* Large number of intermediate stages requires excessive documentation.

# V-MODEL

The V-model is a type of SDLC model where process executes in a sequential manner in V-shape. It is also known as Verification and Validation model. It is based on the association of a testing phase for each corresponding development stage. Development of each step directly associated with the testing phase. The next phase starts only after completion of the previous phase i.e. for each development activity, there is a testing activity corresponding to it.

The V-Model is a software development life cycle (SDLC) model that provides a systematic and visual representation of the software development process. It is based on the idea of a “V” shape, with the two legs of the “V” representing the progression of the software development process from requirements gathering and analysis to design, implementation, testing, and maintenance.

**The V-Model is a linear and sequential model that consists of the following phases:**

1. Requirements Gathering and Analysis: The first phase of the V-Model is the requirements gathering and analysis phase, where the customer’s requirements for the software are gathered and analyzed to determine the scope of the project.
2. Design: In the design phase, the software architecture and design are developed, including the high-level design and detailed design.
3. Implementation: In the implementation phase, the software is actually built based on the design.
4. Testing: In the testing phase, the software is tested to ensure that it meets the customer’s requirements and is of high quality.
5. Deployment: In the deployment phase, the software is deployed and put into use.
6. Maintenance: In the maintenance phase, the software is maintained to ensure that it continues to meet the customer’s needs and expectations.
7. The V-Model is often used in safety-critical systems, such as aerospace and defense systems, because of its emphasis on thorough testing and its ability to clearly define the steps involved in the software development process.



**Verification:** It involves static analysis technique (review) done without executing code. It is the process of evaluation of the product development phase to find whether specified requirements meet.

**Advantages:**

* This is a highly disciplined model and Phases are completed one at a time.
* V-Model is used for small projects where project requirements are clear.
* Simple and easy to understand and use.
* This model focuses on verification and validation activities early in the life cycle thereby enhancing the probability of building an error-free and good quality product.
* It enables project management to track progress accurately.
* Clear and Structured Process: The V-Model provides a clear and structured process for software development, making it easier to understand and follow.
* Emphasis on Testing: The V-Model places a strong emphasis on testing, which helps to ensure the quality and reliability of the software.
* Improved Traceability: The V-Model provides a clear link between the requirements and the final product, making it easier to trace and manage changes to the software.
* Better Communication: The clear structure of the V-Model helps to improve communication between the customer and the development team.

**Disadvantages:**

* High risk and uncertainty.
* It is not a good for complex and object-oriented projects.
* It is not suitable for projects where requirements are not clear and contains high risk of changing.
* This model does not support iteration of phases.
* It does not easily handle concurrent events.
* Inflexibility: The V-Model is a linear and sequential model, which can make it difficult to adapt to changing requirements or unexpected events.
* Time-Consuming: The V-Model can be time-consuming, as it requires a lot of documentation and testing.
* Overreliance on Documentation: The V-Model places a strong emphasis on documentation, which can lead to an overreliance on documentation at the expense of actual development work.

# RAD MODEL

# RAD which is abbreviated as Rapid Application Development Model, is based on the concepts of both iterative and prototyping development model. The Rapid Application Development model basically take prior attention on assembling customer desires in the course of workshop and development, and there is an emphasis on early testing of the system's prototype by the client or consumer using iterative methodology, then if based on the feedback - reuse the existing prototype(s) and stays in constant assimilation as well as rapid delivery. Here, in this model, the modules and pieces, as well as various functions, are developed in parallel in the form of mini projects to save the development time. These models and functions are boxed together, delivered and further assembled to form a working prototype.

The five stages of rapid application development (RAD) model are:

1. **Business Modeling Phase**: The processed data flow is recognized from varieties of business perspective.
2. **Data Modeling Phase**: Information that is taken from business modeling is then implemented for defining elements which are required for the business.
3. **Process Modeling Phase**: Data modeling achieved after assimilating details from business information flow needs to be appropriately processed to come up with a prototype.
4. **Application Generation Phase**: Various autonomous tools are employed for converting process models to code which is then finally converted to the actual system.
5. **Testing and Turnover Phase**: All interfaces and modules are tested.



## Benefits of Using RAD model

* Reusability of components makes or speeds up the development and reduces the time that it needs for developing a product.
* The modularized way of crafting each function within the system makes the development task easier.
* Large projects can be done easily through the RAD model.

## Drawbacks of Using RAD model

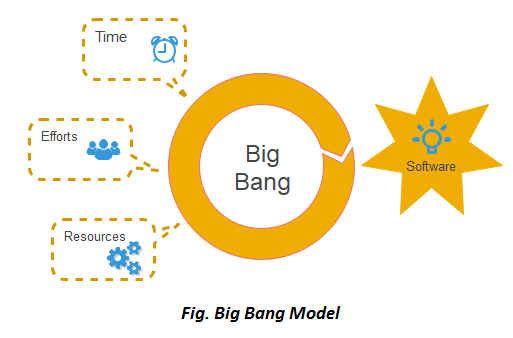
* A proper time-frame should have to be maintained for both end customer as well as developers for completing the system.
* RAD model-based software development fails because of a lack of commitment and dedication.
* A slight complexity in the modularizing in RAD model can lead to failure of the entire project.

# 

# BIG BANG MODEL

In this model, developers do not follow any specific process. Development begins with the necessary funds and efforts in the form of inputs. And the result may or may not be as per the customer's requirement, because in this model, even the customer requirements are not defined.

This model is ideal for small projects like academic projects or practical projects. One or two developers can work together on this model.



## **When to use Big Bang Model?**

As we discussed above, this model is required when this project is small like an academic project or a practical project. This method is also used when the size of the developer team is small and when requirements are not defined, and the release date is not confirmed or given by the customer.

## **Advantage(Pros) of Big Bang Model:**

1. There is no planning required.
2. Simple Model.
3. Few resources required.
4. Easy to manage.
5. Flexible for developers.

## **Disadvantage(Cons) of Big Bang Model:**

1. There are high risk and uncertainty.
2. Not acceptable for a large project.
3. If requirements are not clear that can cause very expensive.