* Difference between procedural and object-oriented programming language



Table

Description automatically generated with medium confidence

* SSADM (Structured Systems Analysis and Design Method)

SSADM is applicable to projects with technical and business requirements that are unlikely to change. It uses a top-down [waterfall](https://www.techtarget.com/searchsoftwarequality/definition/waterfall-model) project management approach to information systems projects. SSADM isn't suitable for projects in a volatile business environment, so [Agile](https://www.techtarget.com/searchsoftwarequality/tip/Waterfall-vs-Agile-vs-iterative-development-explained) should be used instead.

* Difference between waterfall and agile methodology

Diagram

Description automatically generated

In a waterfall model, each phase must be completed before the next phase can begin and there is no overlapping in the phases.

The Waterfall model is the earliest SDLC approach that was used for software development.

The waterfall Model illustrates the software development process in a linear sequential flow. This means that any phase in the development process begins only if the previous phase is complete. In this waterfall model, the phases do not overlap.

Some situations where the use of Waterfall model is most appropriate are,

* Requirements are very well documented, clear, and fixed.
* Product definition is stable.
* Technology is understood and is not dynamic.
* There are no ambiguous requirements.
* Ample resources with required expertise are available to support the product.
* The project is short.

Diagram

Description automatically generated

In the Agile method, the entire project is divided into small incremental builds.

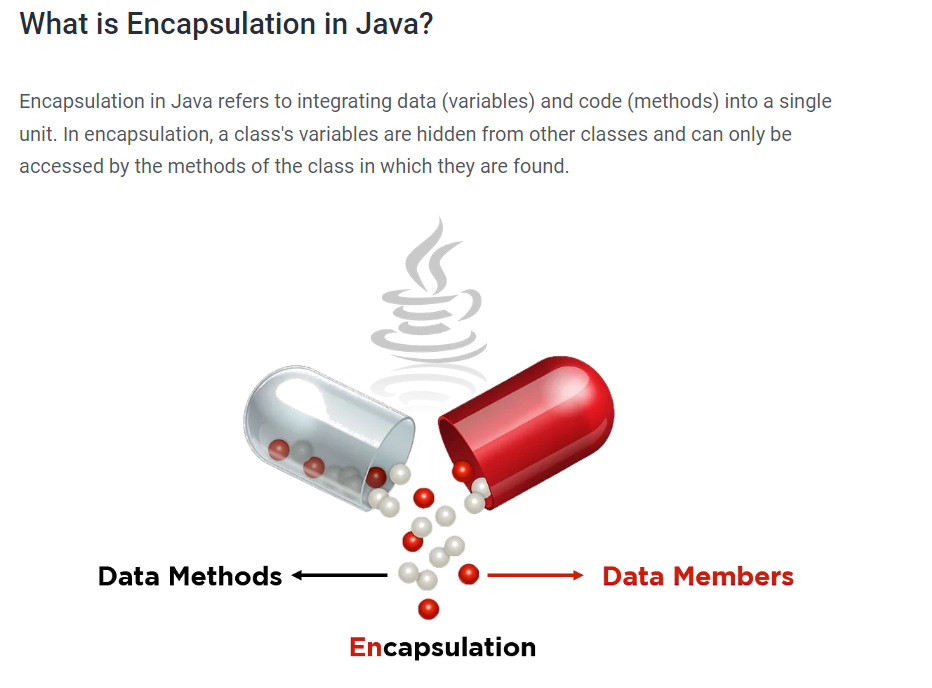
All these builds are provided in iterations, and each iteration lasts from one to three weeks.

They are very much visible in organizations. There are some of the important points related to the agile model listed as follows,

* Agile focuses on customer feedback, collaboration, small and rapid releases.
* Its purpose is to manage complex projects.
* The Agile produces better application suites with the desired requirements. Moreover, it can quickly adapt according to the changes made on time during the project life.
* It has a small team size. Therefore, fewer people work on it so that they can move faster.
* The agile model is not a suitable model for small projects. The expenses of developing the small projects using agile are more than compared to other models.
* In agile methodology, the interaction of customers is very high, as after each iteration an incremental model is deployed to customers.
* Key concepts of Object-Oriented Programming

**Abstraction**:

Data abstraction is the process of hiding certain details and showing only essential information to the user. Abstraction can be achieved with either abstract classes or [interfaces](https://www.w3schools.com/java/java_interface.asp).



Graphical user interface, text

Description automatically generated

**Inheritance**:

In Java, inheritance means creating new classes based on existing ones. A class that inherits from another class can reuse the methods and fields of that class. In addition, you can add new fields and methods to your current class as well.

**Polymorphism**

Which literally means “different forms,” is one of the core concepts of OOP. Polymorphism explores how to create and use two methods with the same name to execute two different functionalities — like adding two functions with the same name but that accept different parameters.

* Objects:

Real-world objects share two characteristics:

They all have state and behaviour. Dogs have state (name, colour, breed, hungry) and behaviour (barking, fetching, wagging tail). Bicycles also have state (current gear, current pedal cadence, current speed) and behaviour (changing gear, changing pedal cadence, applying brakes).

Software objects are conceptually similar to real-world objects: they too consist of state and related behaviour. An object stores its state in fields (variables in some programming languages) and exposes its behaviour through methods (functions in some programming languages).

* Relationship:

A relationship in Java means different relations between two or more classes. For example, if a class Bulb inherits another class Device, then we can say that Bulb is having is-a relationship with Device, which implies Bulb is a device.

In Java, we have two types of relationship:

1. Is-A relationship: Whenever one class inherits another class, it is called an IS-A relationship.
2. Has-A relationship: Whenever an instance of one class is used in another class, it is called HAS-A relationship.