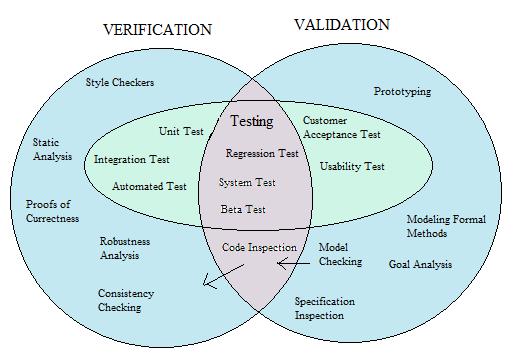
**Verification and validation**:

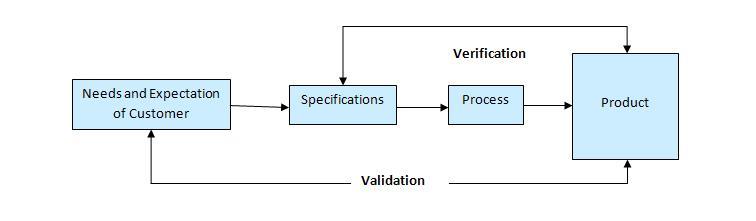
In software engineering , verification and validation (V & V) is defined as the process of confirming that the application meets the requirements and customer specifications, and properly fulfills its purpose.

This process, as the name implies, it is divided into two parts, although they are complementary:



**Software verification:** The process of evaluating software to determine whether the products of a given development phase satisfy the conditions imposed at the start of that phase.

**Software validation**: The process of evaluating software during or at the end of the development process to determine whether it satisfies specified requirements.



**Software testability**:

Software testability is the degree to which a software artifact (i.e. a software system, software module, requirements- or design document) supports testing in a given test context. If the testability of the software artifact is high, then finding faults in the system (if it has any) by means of testing is easier.

The testability of software components (modules, classes) is determined by factors such as:

**Controllability**:

It determines the work it takes to set up and run test cases and the extent to which individual functions and features of the system under test can be made to respond to test cases.

In pdfmake the degree of controllability of test cases are high. For example; columnCalculator.js,printer.js can be controlled.

**Observability:**

It determines the work it takes to set up and run test cases and the extent to which the response of the system under test (SUT) to test cases can be verified.

In pdfmake all the test cases like printer.js,columnCalculator.js,fontProvider.js etc were working.

**Isolateability:**

It determines the degree to which the component under test (CUT) can be tested in isolation.

In pdfmake the degree of isolateability varies with every test cases.for example: printer.js has very less degree of dependency so it can be tested separatly.Although it too has some dependencies but they are very low and easily controllable. Test like docMeasure.js depend on styleContextStack.js so the degree of dependency is high.so,it is not feasible to test them separately.

**Separation of concerns:**

It determines the degree to which the component under test has a single, well defined responsibility.

In pdfmake the degree of responsibility of test is single and well definied, as for example;

documentContext.js: it keeps the page orientation fixed while moving to the next page. layoutBuilder.js: It gives maxmimum height option to text. printer.js: It prints the pdf on client side.

**Understandability:**

It determines the degree to which the component under test is documented or self-explaining.

In pdfmake some of the test cases are self-explaining while for some the understandability increases with the dependency. For example; printer.js is self-explaining, while docMeasure.js and styleContextStack.js have some dependency and are understandable when they are read together.

**Heterogeneity:**

It determines the degree to which the use of diverse technologies requires to use diverse test methods and tools in parallel.

In pdfmake the degree of heterogeneity is very less,still it uses one technology that is: Npm (node package manager):It is a package manager for JavaScript.