Perfect software doesn’t exist. Software is constantly improving and evolving. There is continuous patches/updates made to software.

**Software quality dilemma**

If software is bad, who will want to use it/but it?

While you cannot make software perfect, you can make it “**good enough”.  
-**Delivering a useable product with known defects, bugs, but with the promise that the issues will be ironed out as you go on.  
-Not all features might be available but solves main problem.

**What is Quality**

Quality is very difficult to describe.

**Transcendental view**: Something you can see, but very difficult to explain.

You have to see quality from the **user’s perspective**.  
-Look at how it’s going to be used.

**Manufacturing Quality View:** The original specification and how much it conforms to that specification.

**Value based view**: Are you willing to buy the product.  
-Measures quality based on how much people are willing to pay for it.

**Quality is:**

Quality is defined by multi-factors.

-Transcendental  
-User  
-Manufactures  
-Product  
-Value-Based

Quality of design refers to the characteristics that designers specify for a product

Quality of conformance focuses on the degree to which the implementation follows the design and the resulting system meets the requirements and performances goals

-User science = compliant product + good quality + delivery within budget and schedule

-A product quality is a function of how much it changes the world for the better

-If software provides substantial benefits to end users, they may be willing to tolerate occasionally reliability or performance issues.

**Software quality**

An effective software process applied in a manner that creates a useful product that provides measurable value for those who product it and those who use it

-Effective software …

-Useful process

- …