Rochak Kunwar

Week 01

1. Accuracy: As we all know, accuracy is something that we achieve with high precision and trueness. There is a huge competition around us and everything we do must give high precision than our competitors. Failing to do so might cause the reason to fade out from the market.

For this metric, we can try different approach for testing if our software is accurate as it is supposed to be. We all in this class have used the BYUI website and curriculum page of our department. We can easily see some information on the web page that are outdated and are not updated. I don’t know who exactly do that type of small changes, but I believe it is done by web department in campus.

For checking accuracy of the BYUI website, we can select 16 participants, 4 from each school year and all 4 of one school year from different background. We can ask them to see the curriculum for their major. We can check how fast and how accurate results they get from the web site. A lot of pdf file that we get straight from google are old and outdated. Those files, I don’t know why are not deleted. Maybe those are useful for old students. But there is no big bold warning/indicator to indicate the new and old curriculum. If we pull out the number of people that will get the right file no matter the source and time, they take to get the right file then we can compare it to the total number of participants. If the percentage of success is low then we can conclude that the design of ‘curriculum pdf file’ in itself must be changed so there is warning with high visibility indicating the year.

This method is very valid, and reliable because we are testing the file straight. We might not be checking the software, but it is a very important part that software’s/web-app should focus on providing updated results. As well as it is an Efficient test because we are testing in fewer people and one time.

1. Availability: It is defined as how much of the function in software is available when we need it. We can relate this to getting a super expensive car where some function doesn’t work but it is totally drivable, and you don’t even notice the problem. Different software has different type of availability. Like our autonomous car can have very less availability in certain situation because of an ethical problem.

An autonomous car can do lot of things it can avoid an accident, but what if the car is in high speed or it is in the highway. How it will react will those collision avoidances would be activated or not. Of course, these things are coded by human so it will behave according to the instruction but there is a chance of doing totally different in testing phase of this software. It might not do what we thought it will do. We can check this by putting different virtual obstacles and test it. By using virtual method, it will be very efficient, and repeating number of times will make it reliable, and since there is not danger of life in virtual test it will be a valid to do so.

1. Efficiency: From the research paper, “a review of software quality models for the evaluation of software products” efficiency is defined as “The degree to which the software product provides appropriate performance, relative to the amount of resources used, under stated conditions.” Efficiency is most important metric in mobile software and fairly important in other software too.

We can test this metric by letting user use the software. We can go out and ask few people in public, or someone related for whom we are creating a software. If it is a mobile app, we can ask people on the street or people we know. If it is a big software, we can ask tester in those fields to test our software’s efficiency. I feel like the best way to test efficiency will be to compare your software with different software in the market parallelly. We can create a category of software we want to compete with and test for only those. That way we will be saving our time and money also it will give us what we have better or worse than our competitor.

1. Reusability: From the research paper, “a review of software quality models for the evaluation of software products” reusability is defined as “The degree to which an asset can be used in more than one software system, or in building other assets.” I am personally a big fan of this. It’s a lot of work when we have to redo a thing which is already done. Reusability allows us to progress even better and try different new things using old things we have. It also helps in maintaining replaceability of the code or software parts.

I feel like this metric is used even when we are developing the software from the scratch. When different developers are working on same project and they see some new version in the git after a week and lot of things have been changed. Then, in some way it is just like using the old code to do something more. I think this metric is very easy to test. We can just keep couple of developer from our workgroup and just ask them to figure out what the code is doing. We don’t have to even write or do anything additional, only we have to do is understand what the code is doing. Understandably helps a lot in making code reusable. If the group of developers can figure out what the code is actually doing then we can conclude the code is reusable which will make a whole software reusable.

Reference:

[1]. J. Miguel et al. "A review of software quality models for the evaluation of software products," *International Journal of Software Engineering & Applications*, vol. 5, no. 6, pp. 31-54, Nov. 2014,  
[Online] Available: [http://airccse.org/journal/ijsea/papers/5614ijsea03.pdf](https://content.byui.edu/items/8bcbc45e-012a-48e0-800d-7082dd962f15/1/?.vi=file&attachment.uuid=71271bff-7553-4c47-b501-23c03a545819)