**Brandon Hobbs**

**CS-320**

**June 28, 2022**

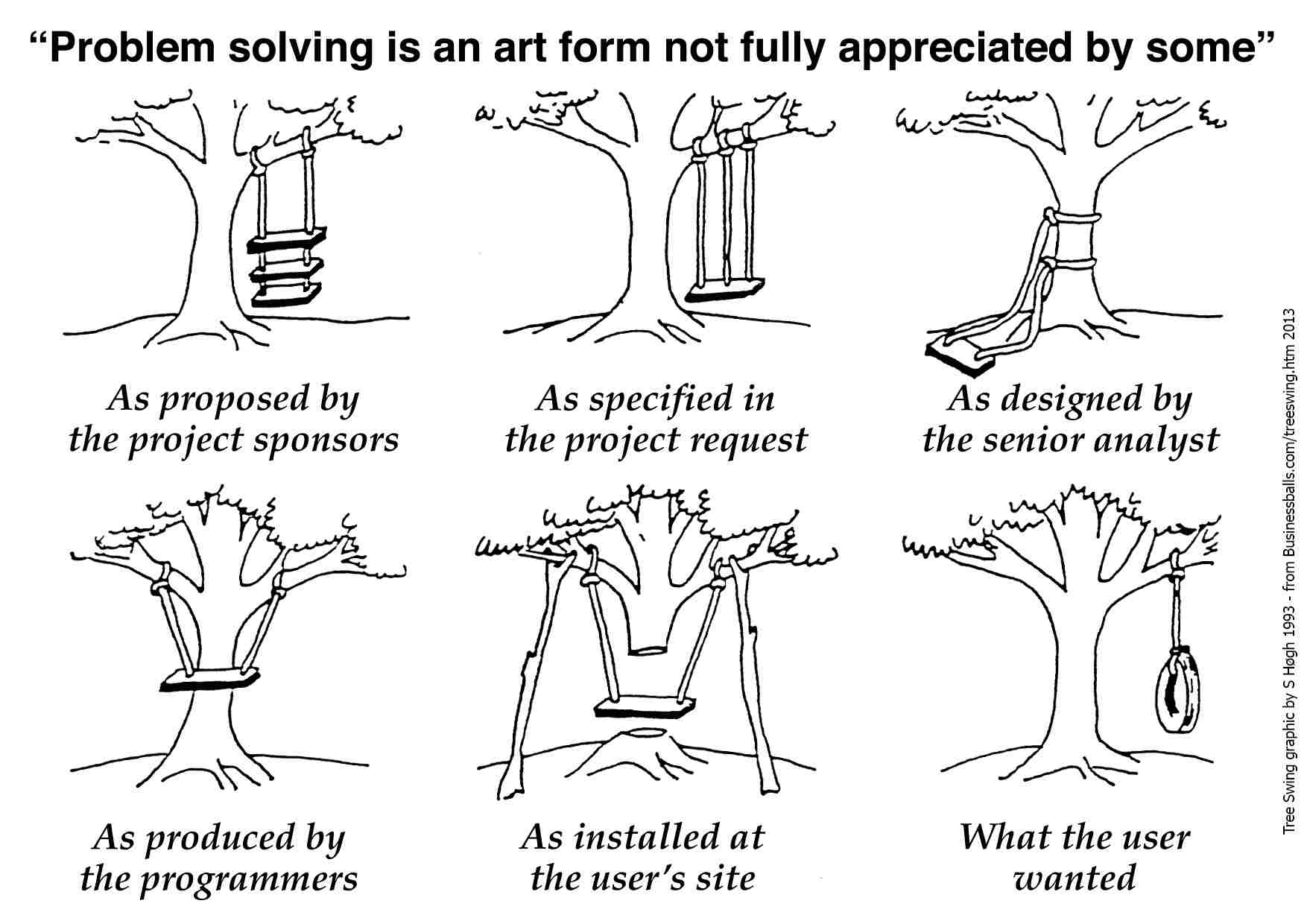
**Journal Module 1**

The software development life cycle includes the different steps needed to identify user need, plan the work, do the work, and then ensure that creation is correct. These steps are often seen as distinct and separate process. I do not believe they are – not even when testing is conducted by a separate team.

Testing has 2 flavors: verification and validation. Verification checks for "bugs" by looking at the specifications. Validation makes sure we build the right thing by asking the user. We verify by testing. We validate with user acceptance testing.

This verification (“testing”) should happen throughout the SDLC. If, as with waterfall, testing is related to the final step of the project there is a great chance to have a failure. This failure, had it been caught earlier, might not have been as big an issue.

Validation is also essential. A piece of software is generally written to fill a need – usually for profit. Even if the software is free of bugs, i.e., it meets all of the specification, the specifications could have been misunderstood. I’ve always thought about validation in terms of that famous tree swing comic (Chapman, 2021):



Validation is asking the user how he likes his tire swing.

Validation can be carried out through the entire SDLC also. Often this is done by having domain experts within the team. They take on the voice of the user and fight for the user’s needs and not for the developers’ needs.

Lastly, testing needs to be planned. Many times, the code needs to have “hooks” built into it so that there are test points, e.g., have your algorithm write all the loop variables into a txt file for error checking.

We tend to say we want to test left. That is, we want to plan way ahead for testing and validation to prevent catastrophic surprises at the end of a project. There is one time when testing must wait until the end – safety.

There are embedded systems that cannot be tested safely until enough of the system is built to allow that testing to happen. In this case, we often use techniques such as software-in-loop or hardware-in-loop testing. This is similar to top-down approaches utilizing function stubs but with simulators taking the place of actual hardware components.

I do not accept customer availability as a reason to delay testing. Domain experts can act as a proxy to the customer. Domain experts are generally expensive but prove to be cheaper than a catastrophic error right before release.

Chapman, A. (2021, October 4). *Tree swing cartoon pictures (early versions).* BusinessBalls. https://www.businessballs.com/amusement-stress-relief/tree-swing-cartoon-pictures-early-versions/