Portfolio III

Theodor Baur

17/05/2022

—

CM1301: Principles, Tools, and Techniques for Secure Software Engineering

Section 1: Reflection on Class Material

This week I rewatched the video "Rapid Prototyping: Sketching | Google for Startups" (Google for Startups, 2016). This outlined the three methods of prototyping that could be used, sketching and paper prototypes, digital prototypes and native protoypes, with each method having a unique place and function.

Sketching is flexible and can be used at any stage of the design process, for example at the start simple "wireframe" sketches can be used to abstract design ideas down to their simplest form, creating a clear structure which can be easier to conceptualize and allowing multiple basic ideas to be drawn quickly. They are also used at later stages in the design process to create details such as button and color designs once the bulk of the design is finished. In my view, the main benefit of sketching is how quick and easy it is to draft a multitude of designs. Any time a change is suggested you can see how the change looks quickly before implementing it digitally, which takes more time.

Personally I used a form of sketching to plan the structure of my *e-commerce* website for CM1102 project. I took a departure from traditional sketching and drew my *wireframe* plans using MS Paint but the underlying benefits were the same. I would analyse the layout of other *e-commerce* websites such as *Amazon, Asos* and *Nike* by sketching the structure, including where the tiles were positioned, the contents of the navigation bar, the scrolling direction and any drop-down menus to help inform my own design. Once I was relatively sure of my basic design I would start to try and implement this structure in my HTML and CSS work. I contrasted this to some other design work that I had done for the CM1102 module, building a website about *Alan Turing,* where I jumped straight into design instead of sketching initially and recalled that the sketching helped me build the website much quicker, with a much more intuative design. I am keen to implement more sketching into more coding and design projects in the future.

Section 2: Independent Research

For this semester I decided to read "A Plea for Lean Software" *(Wirth, 1995)*, a paper describing the relationship between the performance cost of software and the actual performance of software over time. The main ideas that Wirth puts out are that hardware tehcnology is improving extremely quickly, therefore software requirements expand to fill the availiable memory, however the software is getting slower at a higher rate than hardware becoming faster.

Wirth suggests that the increasing complexity of sofware is the reason software is becoming slower and more demanding. Wirth grants that software must become more complex to solve complex problems that we have today, however he believes that there is unnecessary complexity and the main reason for this is because customers don't know difference between essential features and "nice-to-have" features, claiming that developers often add any feature the user wants uncritically. In reality each customer will only use a small number of the given features, ideally we would have a basic design and modular extenstions so there will be less *nice-to-have* features included in any base build.

He also states that time pressure can produce more unnecessarily complex software as it discourages planning and improvements. This is because often in order to meet market deadlines, quickly-made software additions and corrections are favoured. Time pressure exists as often the first product on the market is more successful than the following products, even if they are better designed, and often the first product is established as the de-facto standard.

From reading this paper my approach has changed towards features. Before I considered any *nice-to-have* as a net positive to my own software, however this helped create a more balanced view as the increased complexity brought about by unnecessary features can make my code bloated and unreadable. Instead I would like to take a modular approach to any future software so I can focus my efforts on creating more effective essential features.

Appendix

- Google for Startups. 2016. *Rapid Prototyping: Sketching | Google for Startups*, Available at: https://www.youtube.com/watch?v=JMjozqJS44M [Accessed 17 May 2022]

- Wirth, N. 1995. *A Plea for Lean Software,* Available at: https://blog.frantovo.cz/s/1576/Niklaus%20Wirth%20-%20A%20Plea%20for%20Lean%20Software%20-%20OCR.pdf/ [Accessed 17 May 2022]