Khashayar Etemadi

April, 2021

FDD3001 VT21-1 Research: Theory, Method, Practice

What is “understanding” in software engineering?

When we think about the notion of “understanding”, we can contrast it with the related notion of “knowing”. If I understand that *P*, I necessarily know that *P*. On the other hand, if I know *P*, I may not understand that *P*. So, one can ask what is the difference here? The difference is that to understand something, you should be able to put it in the context of a big picture [1]. Consequently, you should be able to explain a phenomena to understand it, while this is not the case for knowledge.

The next question that comes to mind is this: “what does it take to be able to explain something?” This question can be answered in many different ways [2], but I believe the core of any valid explanation is a clarification of the causal chains involved. For example, to explain how vaccination leads to immunity, we should clarify what are the causal effects of vaccination and how their final result is immunity against a certain disease.

Now we can go back to the comparison between “knowing” and “understanding” that we started this short essay with. To know that *P*, it is sufficient if you have a justified true belief [3] that *P*. On the other hand, *to understand that P, you should know that P and also know the causal chains linked to P*. For instance, many people with very little knowledge about physics may know that nothing can travel faster than light, but only people familiar with Einstein’s relativity theory understand this proposition because it is the theory that gives us the causal chains behind this phenomena.

If we accept the aforementioned definition of understanding, then we can readily explain the difference between understanding in various fields of science. Each field of science studies a certain type of causal relations. Quantum physicists study causal relations between subatomic particles; whereas, psychologists study causal relations between states of minds. Therefore, to understand a murder case from a psychological point of view, you should know the murderer’s states of mind that led to the decision to kill the victim. In contrast, to understand the physical aspect of the very same event, one should know the relations between physical objects that led to the victim’s death.

We can now discuss specific features of understanding in my research area, software engineering. Per wikipedia definition [4]: “A software engineer is a person who applies the principles of software engineering to the design, development, maintenance, testing, and evaluation of computer software.” As you see, this definition implies that a software engineering scientist studies causal relations between software programs, their design, development, maintenance, testing, and evaluation.

In my own research, I try to improve the maintenance of software programs. Currently, we are working on two types of tools. First, tools that help developers avoid crashes in their programs, and second, tools that automatically fix bugs in programs whenever they crash. To design such tools, we have to “understand” what causes a crash in a program. For this purpose, we study the frequent wrong decisions that programmers make while developing a software and warn them when they are repeating such mistakes. We also need to “understand” what causes a fix in a program. To this end, we investigate the common patterns that human developers follow to fix programs and simulate such patterns in our *automatic program repair* tools.

References:
[1]
https://qz.com/1123896/its-better-to-understand-something-than-to-know-it/#:~:text=%E2%80%9CKnowing%E2%80%9D%20and%20%E2%80%9Cunderstanding%E2%80%9D,to%20form%20a%20big%20picture
[2] https://plato.stanford.edu/entries/scientific-explanation/
[3] https://plato.stanford.edu/entries/knowledge-analysis/