Software Engineering – Stephen Barrett

Thomas Donovan – 14316040

Margaret Hamilton

Margaret Hamilton is one of the most famous women in software engineering, in fact she is even credited with coining the term software engineering while working for the National Aeronautics and Space Administration in MIT’s Instrumentation Laboratory, now named the Charles Stark Draper Laboratory. She started using the term as she believed her work and the work of her team was not being taken seriously enough and through their work and the methods they used and standards they put in place they gained the respect and recognition of their peers in other engineering disciplines.

While coming up with the very name of software engineering seems a large enough, her real contribution was her methodology and dedication, which helped gain the respect of other engineers and scientists, her work on the Apollo program, which set mankind on the moon and earned her the presidential medal of freedom, and her contributions through higher order software and her Universal Systems Language and “development before the fact” paradigm.

Originally Margaret Hamilton did not plan on becoming a software engineer, in fact she started on that path by taking an interim position at MIT which eventually led to her work on the SAGE project, an early weather tracking system and the Apollo program. Luckily for us all she remained working for NASA and was instrumental in putting the first man on the moon. her brilliance meant that she quickly rose to become the lead software developer for the lunar and command module flight software. From there she co-founded “Higher Order Software” a company that specialized in using the knowledge they had gained in the Apollo program in other software projects, and also eventually Hamilton technologies, which developed her Universal Systems Language.

While working on the Apollo program she learned the importance of avoiding and managing errors early on, preventing them from ever being able to occur or making sure that if they did it was possible to work around them, an excellent example of this was a known error where if a prelaunch program known as P01 was launched midflight it would cause issues, she wanted to program around this error to prevent it from ever being able to occur, however it was assumed that the mistake would never be made and so was ignored, as it turns out it did happen during the Apollo 8 mission, but fortunately they were able to work around it. The thinking was that the user would not make the error, however, as has become clear in recent software development, often if it can be broken, the user will break it, whether it makes sense or not. Due to this she believed in “man in-the-loop” programming, where the software would keep the user updated on any errors so that they could take appropriate action, this was what prevented Apollo 11 from being aborted after an issue with an aerial caused too much information to be sent to the computer 3 minutes before the lunar module was due to land on the moon. the mission was saved because the computer was smart enough to realize that it had been asked to do too much and so was prioritizing the most important task, which in this case was landing on the moon.

Her work was made more impressive by the immense pressure she was under and the grueling process by which they programmed. They programmed with punch cards and the program was then literally woven together through magnetic coils, if the wire went through that was a 1 and outside the coil was a 0. Also, any mistake that was made could cost billions of dollars and the lives of the 3 astronauts on the shuttle. she did all this while minding her young children, who would reportedly sometimes accompany her to the labs and entertain themselves around the corridors of NASA while she worked on landing men on the moon. The actual code used in the Apollo 11 program was uploaded to GitHub by an ex-NASA intern and can be found [here](https://github.com/chrislgarry/Apollo-11).

Her work after the Apollo program included co-founding Higher order software, which provided software that the military used in many projects, and then she eventually broke away and formed her own Hamilton technologies, whose main project has been the development of the universal systems language and the 001-tool suite. The language was developed to prevent errors from ever happening, or at least making them obvious early in the projects life cycle, which saved time and effort and avoided expensive testing.

Her philosophy of software development focused on the reliability of software, and how in its reliability it could avoid excessive testing, handle errors instead of remaining unaware them or cause a complete crash of the system. She also believed that software should tell the user what is happening and what to do so that complete failure can be avoided and the software can continue running. These philosophies were driven by her work on such critical systems and she strongly believed in doing things right the first time, which seems to have been abandoned in today’s world of push now, fix later development. I believe that many of her ideas and philosophies could be useful to modern software development, where well over half of software projects fail. It seems very likely that we could have much to learn from the programmer who helped get mankind to the moon and back.