Abraham Odukoya

Margaret Hamilton

Coiner of the term “software engineering”, Margaret Hamilton was one of the earliest influences on the field. Known for her work on NASA’s Apollo space missions, which landed the first men on the moon, she helped to bring credibility to the emerging discipline and establish it as a respected field in its own right.

Born Margaret Heafield on August 17, 1936 in Paoli, Indiana, USA, Hamilton studied mathematics and philosophy at Earlham College in Indiana, graduating with a bachelor’s degree in 1958. She briefly became a high school mathematics teacher, before moving to Boston with her husband James Hamilton. Although originally planning to begin graduate study in abstract mathematics at Brandeis University, Hamilton accepted what was supposed to be a temporary position at the Massachusetts Institute of Technology (MIT) instead. It was at MIT that she began developing software; her first role involved programming software that could predict the weather on MIT’s Librascope LGP-30 and PDP-1 computers.

She continued her foray into software development by working on the US SAGE air defence system, one of the first of its kind. As an early programmer, Hamilton had little previous instruction or literature to fall back on. She and her team had to develop best practices and approaches for themselves as they worked, and this experience helped her to formulate her ideas on software engineering. The reliability of software would become a key focus of Hamilton’s throughout her career, as she often worked on critical systems where software crashes could have fatal consequences.

Hamilton moved onto her next, and most famous, project during the 1960s: NASA’s Apollo space missions. After joining the team she was soon appointed the lead programmer, as her prodigious talent for coding and designing complex systems became appreciated. Hamilton and her team were responsible for creating the on-board flight software for command and lunar modules, which featured guidance and control systems. This behemoth of a software system would involve over 350 engineers and take 1400 man-years of programming. Software development projects of this size and scale were a novelty at this time, and appropriate techniques for managing them had not yet been developed. The challenge facing Hamilton was not just how to make the software, but what approach to take when making it. Like the astronauts she was writing software for, Hamilton was at the cutting-edge of her field.

Without having the body of work on software engineering that we have today Hamilton and her colleagues found themselves creating new software engineering techniques as they programmed, with any novel problems necessitating new techniques. Hamilton developed the “man-in-the-loop” concept to allow the astronauts to interact with the software during a flight so that high priority tasks would be shown on the astronauts’ displays in real-time. As the success of a space mission depended heavily on the efficacy of the flight software, Hamilton’s code had to be impervious to bugs and errors, leading her to develop strict testing and code analysis regimes. NASA bureaucracy, however, often stifled her innovative solutions to issues in the software.

One error she found and attempted to mitigate was the loading of incorrect programs into the computer mid-flight, which would cause the system to lose its navigation data. Her superiors refused to allow her to add more software to fix the issue, worried that more code would lead to more bugs and an unreliable system. They felt safe in the assumption that no astronaut would be foolish enough to load the incorrect program during a mission. Naturally, during the Apollo 8 mission, Hamilton was proven right and astronaut Jim Lovell loaded the incorrect program causing the computer to lose its navigation data. Working from the ground, Hamilton's team tirelessly studied their software to find a method of recovery which was fortunately successful. Hamilton’s approach to the as yet unnamed field of software engineering was proving fruitful.

Another feature that she developed that would prove crucial to landing a man on the moon was a priority task scheduler, a concept well known to modern operating system developers. Recognising the possibility of the computer being overloaded with tasks, she created the priority scheduler to prevent it from crashing. During the Apollo 11 mission as the moon landing neared the on-board computer nearly ran out of resources, which could have led to its crash and a fatal end to the mission. Hamilton’s foresight meant that the system could prioritise the crucial tasks relevant to the landing and save the mission, proving the efficacy of her programming techniques and the relevance of software engineering as a field. This type of scheduling is found in many software systems nowadays, as it allows programs to run at their optimum potential. Without Hamilton’s work modern software may not have achieved its ubiquity as quickly as it has.

Despite the clear importance of software to the Apollo space missions, the programmers weren’t considered to be in the same league as their colleagues in the other engineering departments. Programming was not considered a true science, or even a discipline in its own right. To combat this stigma around her profession, Hamilton created the term “software engineering” to refer to the work she and the other programmers did. Her aim was to give the field the respect it merited and to put herself and other programmers on an equal footing with the other engineers. She felt that this name would garner the esteem of other disciplines. Initially laughed off as a joke even by her own co-workers, the term gradually came to be accepted as the field developed and is now widespread today. Software engineers are considered just as important as hardware or mechanical engineers to modern technology.

Many techniques that would become staples of software engineering were developed during Hamilton’s time at NASA. After the Apollo missions she and her team performed analysis on the techniques they had used and the lessons they had learned to create a treatise on software engineering as they understood it to be. This lead Hamilton to create the Universal Systems Language (USL), a system-oriented modelling language intended to prevent bugs and errors from being specified in a system’s design. The idea behind USL is that it is better to prevent errors from occurring in the first place early in the development life cycle than to find them by testing later in the development life cycle. It is designed to be preventative instead of being curative, and as such it is very difficult and in some instances impossible to design systems laden with errors. USL bears some resemblance to other formal verification systems used nowadays which are beginning to gain some traction, and can be seen as an early precursor to them.

After her work with NASA, Hamilton left MIT for the private sector. She co-founded the company Higher Order Software in 1976, then founded Hamilton Technologies, Inc., where she is still CEO.

Abraham Odukoya

At Hamilton Technologies, Inc. she has continued work on USL, developing 001 Tool Suite to allow software engineers to turn their specifications of systems in USL to executable programs. While USL might not enjoy the popularity Hamilton would have hoped for, its guiding principles of well specified designs meant to eradicate bugs and errors from the beginning of development is applied to varying degrees by many software engineers, as designing a system thoroughly before attempting to implement it is considered good practice in software engineering.

Hamilton’s contributions to software engineering have been rewarded. She received the NASA Exceptional Space Act Award in 2003 for her efforts during the Apollo space missions and the role she played in putting the first men on the moon. In 2016 she was awarded the Presidential Medal of Freedom by then US President Barack Obama for her contributions to computer science and software engineering.

Margaret Hamilton is known as one of the most important and influential figures in software engineering. Credited with giving the field its name, she also added many important concepts to it, from man-in-the-loop systems to priority scheduling and asynchronous computing. She helped to give the discipline credibility when it was still in its embryonic state. Her approach to engineering software systems, especially the idea of error prevention as opposed to error correction, has left its mark on the industry. Particularly with modern interconnected software systems, which face the constant risk of being hacked or compromised, Hamilton's engineering philosophy of having error-free systems from the design phase right through every stage of development is more relevant, and more important, than ever.

Abraham Odukoya

Bibliography

https://www.britannica.com/biography/Margaret-Hamilton-American-computer-scientist

https://publications.computer.org/software-magazine/2018/06/08/margaret-hamilton-software-engineering-pioneer-apollo-11/

http://www.computerhistory.org/fellowawards/hall/margaret-hamilton/

https://hackaday.com/2018/04/10/margaret-hamilton-takes-software-engineering-to-the-moon-and-beyond/

http://www.htius.com/Articles/r12ham.pdf

https://obamawhitehouse.archives.gov/the-press-office/2016/11/16/president-obama-names-recipients-presidential-medal-freedom