DENNIS RITCHIE

Theodor Barbu
Student no.: 17334387

Structure:

Beginning: Who he is, what he did, why he is important to software engineering, why I chose him for the essay

Content: Very short bio of early life, the beginnings of his career, development of his innovations, challenges faced, direct results of his work, indirect results of his work. results of his work on personal level

Conclusion: Personal opinion

Bibliography: Sources

BEGINNING

Who he is: Dennis Ritchie was an American computer scientist and software developer.

What he did: He is most famous for the implementation of the UNIX operating system, developed alongside Kenneth Thompson, computer scientist, and for his role in the creation and propagation of the C programming language.

Why he's important: The UNIX OS serves as basis for Linux, which is preferred by most big software companies. The C programming language is the basis for the object-oriented programming languages used today. C++, widely used language, is an approach to providing object-oriented functionality with a C-like syntax.

Why I chose him: The reason why I chose Dennis Ritchie for this essay is because I've always taken an interest in the background of programming in general and it was of particular interest to me to read the beginnings of what has evolved into the languages (and the OS, respectively) I and many, if not all, of my colleagues use today. It's even more interesting when I put it in perspective, since these facilities are also used by the vast majority of companies, ranging from the most famous tech giants to the newest start-up in some corner of the world. I view his contributions as arguably the most important in shaping the tech world as we know it.

CONTENT

Short early life bio: He was born in a village in New York in the 1940s, his father was a scientist at Bell Labs and co-author of The Design of Switching Circuits on switching circuit theory.

Beginnings of career: Ritchie got a degree at Harvard in Physics and Applied Mathematics. He also went on to work at Bell Labs on the Multics OS.

Development of innovations: To supplement assembly language with a system-level programming language, Ken Thompson (software engineer, Ritchie's colleague who was working alongside him on the Multics OS) created B. Later, B was replaced by C, created by Ritchie, who continued to contribute to the development of Unix and C for many years. During the 1970s, Ritchie collaborated with James Reeds and Robert Morris on a ciphertext-only attack on the M-209 US cipher machine that could solve messages of at least 2000–2500 letters. It should be noted that he is also known for working on ALTRAN and BCPL.

Challenges faced: The most notable challenge Ritchie faced when creating C was the lack of a fully "general-purpose language" at that time. Assembly language and machine code were the norm and few people would anticipate that people would want to start writing their own software. C managed to bring about the exact changes that were needed, the ones that the earlier Fortran slightly missed. The most difficult aspect was arguably rendering C in such a simple way to understand and use. Most of the programming languages right now, while more versatile in terms of how you can apply them, have a more complex and, as such, difficult to use syntax and variety of features, while C manages to capture the most important aspects of creating a computer program in a

Direct results of work: C is used prominently for systems programming in implementing operating systems and embedded system applications. C can also be used for website programming using CGI (Common Gateway Interface) as a "gateway" for information between the Web application, the server, and the browser. Despite being created in 1972, the need to use it still exists, as it executes faster than other popular programming languages like Java when it comes to system software programming. Many companies with notable profits and impact on the industry have C as their preferred language, which in turn motivates students and aspiring software engineers to have at least have a look at how it works. One other result is that C is one of the languages taught in Computer Science courses, underlining how important it is to understanding the basic principles of programming.

Indirect results of work: C has directly and indirectly influenced several important programming languages, like Python, Java or Perl. The most pervasive influence has been syntactical, all of the languages mentioned combine the statement and (more or less recognizably) expression syntax of C with type systems, data models and/or large-scale program structures that differ from those of C, sometimes radically.

Linux is a Unix-like OS, which has gained a lot of popularity over the past few years. One reason is that it is open-source, which means people can add new features to its kernel and even improve some of the existing ones. It is also very popular because the web interface is considered by many more user-friendly than its counterparts and because of the distribution of processing power.

Personal impact: On a personal level, these changes have enabled me to use the mentioned tools not only during my degree, but also during my spare time, in order to delve into and learn more about programming, see how the most used apps and operating systems are designed and attempt to design small items of my own, activity which would hopefully progress into designing something truly relevant to the world. In other words, these changes have offered aspiring software engineers like myself a platform on which I can express myself and any potential ideas, collaborate with fantastic people and work towards the end goal of producing a certain amount of change which will make different facets of life better.

CONCLUSION

Personal opinion: Dennis Ritchie was undoubtedly a pioneer in the field of software engineering, setting the basis for the development of some of the most useful features of today's tech industry. To sum it up, he changed the way the world works, which is particularly impressive, to say the least. As such, using the word "pioneer" to describe him would not be an overstatement Game and application development is strongly influenced by the innovations brought by him to this area. Citing long time colleague Brian Kernighan, "The tools that Dennis built—and their direct descendants—run pretty much everything today.” Another commentator said, "Ritchie, on the other hand, invented and co-invented two key software technologies which make up the DNA of effectively every single computer software product we use directly or even indirectly in the modern age. It sounds like a wild claim, but it really is true.

BIBLIOGRAPHY

Sources: Wikipedia, Britannica, Quora