**The Portability of C**

In 1983, C compilers were already being built for over 40 different machines, including the most powerful computer at the time, the Cray-1. The increasing popularity of the Unix system demanded an increased concern on portability.

Fortunately, the I/O library and data models provided by C are supported by most operating systems. The issues occur when there's no standardization on the industry relating to, for instance, the ordering of bytes in a word.

The PDP-11, a series of 16-bit minicomputers sold by DEC from 1970 into the 1990s stores bytes from low to high whilst models like the Motorola 68000 microprocessor store from high to low. The size of integers may vary and worse, some machines support 8-bit bytes, others 7 and others 9. Naturally, the portability of a program is compromised if it depends on any of these features.

However, if software isn't directly compatible, it is possible to set properties specific to a particular machine. For instance, if the size of sectors on disk varies from computer to computer, the program can properly handle that variable without the need of rewriting it from scratch. This is the case of UNIX.

**Design decisions**

The language features some wild design decisions which are definitely open for debate, such as floating-point handling or the seemingly random evaluation order in an expression.

**The C programmer**

The C model assumes the programmer makes no mistakes, resulting on sometimes not prohibiting odd constructs. Conversely, Pascal and Ada make it harder to formulate complex instructions and simultaneously detect errors more often.

The article cites C as a language for professionals and that statement still holds as of today. Its freedom allows for truly spectacular mistakes yet its portability and low-level nature are still appealing in the present day.