

A study found that a few simple readability transformations made code shorter and drastically reduced the time to understand it. The Unified Modeling Language (UML) is a notation used for both the OOAD and MDA. This can be a non-trivial task, for example as with parallel processes or some unusual software bugs. Debugging is a very important task in the software development process since having defects in a program can have significant consequences for its users. In the 1880s, Herman Hollerith invented the concept of storing data in machine-readable form. In 1206, the Arab engineer Al-Jazari invented a programmable drum machine where a musical mechanical automaton could be made to play different rhythms and drum patterns, via pegs and cams. The first step in most formal software development processes is requirements analysis, followed by testing to determine value modeling, implementation, and failure elimination (debugging). The first compiler related tool, the A-0 System, was developed in 1952 by Grace Hopper, who also coined the term 'compiler'. He gave the first description of cryptanalysis by frequency analysis, the earliest code-breaking algorithm. As early as the 9th century, a programmable music sequencer was invented by the Persian Banu Musa brothers, who described an automated mechanical flute player in the Book of Ingenious Devices. He gave the first description of cryptanalysis by frequency analysis, the earliest code-breaking algorithm. Debugging is a very important task in the software development process since having defects in a program can have significant consequences for its users. Techniques like Code refactoring can enhance readability. Many programmers use forms of Agile software development where the various stages of formal software development are more integrated together into short cycles that take a few weeks rather than years. Popular modeling techniques include Object-Oriented Analysis and Design (OOAD) and Model-Driven Architecture (MDA). By the late 1960s, data storage devices and computer terminals became inexpensive enough that programs could be created by typing directly into the computers. Techniques like Code refactoring can enhance readability. Assembly languages were soon developed that let the programmer specify instruction in a text format (e.g., ADD X, TOTAL), with abbreviations for each operation code and meaningful names for specifying addresses. When debugging the problem in a GUI, the programmer can try to skip some user interaction from the original problem description and check if remaining actions are sufficient for bugs to appear. Also, specific user environment and usage history can make it difficult to reproduce the problem. Unreadable code often leads to bugs, inefficiencies, and duplicated code. Trial-and-error/divide-and-conquer is needed: the programmer will try to remove some parts of the original test case and check if the problem still exists. It affects the aspects of quality above, including portability, usability and most importantly maintainability. Implementation techniques include imperative languages (object-oriented or procedural), functional languages, and logic languages. Scripting and breakpointing is also part of this process.