3.4: Software Creation

We just discussed different types of software and now can ask: How is software created? If the software is the set of instructions that tells the hardware what to do, how are these instructions written? If a computer reads everything as one and zero, do we have to learn how to write software that way? Thankfully, another software type is written, especially for software developers to write system software and applications - called programming languages. The people who can program are called computer programmers or software developers.

Analogous to a human language, a programming language consists of keywords, comments, symbols, and grammatical rules to construct statements as valid instructions understandable by the computer to perform certain tasks. Using this language, a programmer writes a program (called the source code). Another software then processes the source code to convert the programming statements to a machine-readable form, the ones, and zeroes necessary to execute the CPU. This conversion process is often known as compiling, and the software is called the compiler. Most of the time, programming is done inside a programming environment; when you purchase a copy of Visual Studio from Microsoft; It provides the developers with an editor to write the source code, a compiler, and help for many of Microsoft's programming languages. Examples of well-known programming languages today include Java, PHP, and C's various flavors (Visual C, C++, C#.)

Figure 3.4.1: Convert a computer program to an executable. Image by Ly-Huong T. Pham is licensed under CC-BY-NC

Thousands of programming languages have been created since the first programming language in 1883 by a woman named Ada Lovelace. One of the earlier English-like languages called COBOL has been in use since the 1950s to the present time in services that we still use today, such as payroll, reservation systems. The C programming language was introduced in the 1970s and remained a top popular choice. Some new languages such as C#, Swift are gaining momentum as well. Programmers select the best-matched language with the problem to be solved for a particular OS platform. For example, languages such as HTML and JavaScript are used to develop web pages.

It is hard to determine which language is the most popular since it varies. However, according to the TIOBE Index, one of the companies that rank the popularity of the programming languages monthly, the top five in August 2023 are Python, C, C++, Java, and C# with Julia emerging as a language that is faster than Python (Tiobe, 2023). For more information on this methodology, please visit the TIOBE page. For those who wish to learn more about programming, Python is a good first language to learn because not only is it a modern language for web development, it is simple to learn and covers many fundamental concepts of programming that apply to other languages.

One person can write some programs. However, most software programs are written by many developers. For example, it takes hundreds of software engineers to write Microsoft Windows or Excel. To ensure teams can deliver timely and quality software with the least amount of errors, also known as bugs, formal project management methodologies are used, a topic that we will discuss in Chapter 10.

3.4.1: Open-Source vs. Closed-Source Software

When the personal computer was first released, computer enthusiasts immediately banded together to build applications and solve problems. These computer enthusiasts were happy to share any programs they built and solutions to problems they found; this collaboration enabled them to innovate more quickly and fix problems.

However, as software began to become a business, this idea of sharing everything fell out of favor for some. When a software program takes hundreds of hours to develop, it is understandable that the programmers do not want to give it away. This led to a new business model of restrictive software licensing, which required payment for software to the owner, a model that is still dominant today. This model is sometimes called closed source, as the source code remains private property and is not made available to others. Microsoft Windows, Excel, and Apple iOS are examples of closed-source software.

Many, however, feel that software should not be restricted. Like those early hobbyists in the 1970s, they feel that innovation and progress can be made much more rapidly if we share what we learn. In the 1990s, with Internet access connecting more and more people, the open-source movement gained steam.

Open-source software has the source code available for anyone to copy and use. For non-programmers, it won't be of much use unless the compiled format is also made available for users to use. However, for programmers, the open-source movement has led to the development of some of the world's most-used software, including the Firefox browser, the Linux operating system, and the Apache webserver.

Some people are concerned that open-source software can be vulnerable to security risks since the source code is available. Others counter that because the source code is freely available, many programmers have contributed to open-source software projects, making the code less buggy and adding features, and fixing bugs much faster than closed-source software.

Many businesses are wary of open-source software precisely because the code is available for anyone to see. They feel that this increases the risk of an attack. Others counter that this openness decreases the risk because the code is exposed to thousands of programmers who can incorporate code changes to patch vulnerabilities quickly.

In summary, some benefits of the open-source model are:

- The software is available for free.
- The software source code is available; it can be examined and reviewed before it is installed.
- The large community of programmers who work on open-source projects leads to quick bug-fixing and feature additions.

Some benefits of the closed-source model are:

- Providing a financial incentive for software developers or companies
- Technical support from the company that developed the software.

Today there are thousands of open-source software applications available for download. An example of open-source productivity software is Open Office Suite. One good place to search for open-source software is sourceforge.net, where thousands of software applications are available for free download.

3.4.2: Software Licenses

The companies or developers own the software they create. The software is protected by law through patents, copyrights, or licenses. It is up to the software owners to grant their users the right to use the software through the terms of the licenses. In a later chapter, we will discuss the topic of copyright and licenses in more detail.

Paying for software licensing offers several benefits for individuals, businesses, and organizations, such as receiving technical support, regular updates, and legal protection. Companies usually need the comfort of having regular updates and technical support, even if the software is free.

For closed-source vendors, the terms vary depending on the price the users are willing to pay. Examples include single-user, single installation, multi-users, multi-installations, per network, or machine.

They have specific permission levels for open-source vendors to grant using the source code and set the modified version conditions. Examples include free to distribute, remix, and adapt for non-commercial use but with the condition that the newly revised source code must also be licensed under identical terms. While open-source vendors don't make money by charging for their software, they generate revenues through donations or selling technical support or related services. For example, Wikipedia is a widely popular online free-content encyclopedia used by millions of users. Yet, it relies mainly on donations to sustain its staff and infrastructure.

3.4.3: Reference

TIOBE Index for August 2023. Retrieved August 27, 2023, from tiobe.com