Read pages: 24-27 in the textbook and pay close attention to the material pertinent to Figure 1.3. Write a one page report on the contents and importance of the figure. Create a Word file **fig\_1\_3.docx** to show your answer.

Figure 1.3: The compilation process

The language a compiler translates is called a source language, the process of compiling a program takes place in several phases.

The process begins with the source program then moves onto the Lexical analyzer. This gathers the characters of the source program into lexical units such as identifiers, special words, and operators. This is the step that ignores comments.

The process then moves into the syntax analyzer step. This takes the lexical units and translates them into hierarchical structures called parse trees. These trees represent the structure of the program, what runs first and when things run are decided here. No actual parse trees are built here, just the information for the structure is constructed.

The process moves onto the intermediate code generator step, which builds the code between the primary code and the output machine language. These can look similar to assembly languages. During this process the semantic analyzer checks for errors that could be impossible to detect during syntax analysis.

During the intermediate step, code organization also takes place. This is meant to improve programs by making them smaller or faster. Most optimization is done on intermediate code.

The next step in the process is the code generator. This translates the intermediate code into machine language.

An important part of this process is the symbol table. This table information is placed by the lexical and syntax analyzers and used by the semantic analyzer and code generator.

The last step of the process is the computer, although most machine language can be directly applied to hardware it is usually necessary for there to be supplementary software to bridge the gap between existing software and code a user would make. This is usually found in the operating system. The user and system code together is called the load module, or executable image. A linker (programmer) links system programs and user programs by “linking and loading”.

This concept is very important for a software developer to understand. The steps necessary to go from a code a developer is writing to that code actually doing something is crucial to understanding software engineering as a whole. Knowing how a code is understood by the computer is one of the essential concepts of computer science. Many codes go from a compiler to the hardware in different ways. Many languages are often interpreted in different ways. Understanding how a language is compiled and understood and translated could mean the difference between a successful and unsuccessful code. This is also necessary to develop new coding languages. Software engineers who want to develop code that will go directly to hardware NEED to understand this process. For robots, for operating systems, for independent systems, all of these codes translate directly to software. Now it is different to write something in a java compiler and watch the computer do all the work for you, all you have to do is is write the code and it works or doesn't. But to truly be successful in the field the steps between the creation of a code and the execution of it are fundamental.