CODE ANALYZER

We cannot always create a code which works perfectly fine at all situations. It becomes necessary for us to evaluate the code to exactly point at the part which results in an issue. It becomes tedious to sit and evaluate all the files/pieces of code that constitute a system where we have to keep track of all the types, their dependencies and relationships they hold. Hence, the Code Analyzer becomes extremely important to evaluate the system for system metrics.

The goal here is to build a tool that will help us evaluate a system or a piece of code for the system metrics. To estimate the dependency of one type or token or expression on another and hence help us to highlight the precise files and line numbers that are affected. Also this tool estimates the complexity in implementing the system.

Since this stage comes early in the cycle of SDLC (Software development life cycle), it helps us reduce business and cost risks by identifying vulnerabilities that can cause an issue and helps us increases the overall productivity.

Here the entire system, as mentioned earlier, includes 2 major tasks.

1. Creating tokens and storing them using the right data structures
2. Finding types and type relationships.

The above mentioned tasks can be detailed as follows. Each of the above tasks will be explained in detail in the further sections.

* Parse the command line
* Fetch the files from the path provided
* Analyze the types in files
* Save the type info for the file in a data structure that is efficient such as a tree.
* Continue with the above steps for all the files fetched
* Check if the type examined in the code matches the type and type relationship defined.
* Display the same information onto the screen for further analysis