1. Code base and language picker: Code base is a code of specific language. Language is automatically picked. It can be easily picked using guesslang and additional voting modules since it is not perfect on its own. Code base is assumed to be single language.
2. Entity Graph Generator: It goes through the code base and extracts class names, method names and member names. It builds a tree structure based on the folder structure. This is due to the assumption that test code will be in the same folder as the related sources. So it will allow for searching within the directory structure. Also, I need a secondary structure with just the names, where they have been tagged by type and directory. The user can trigger a structure refresh and also it must refresh itself based on the timestamps of the files.
3. Token Extractor: Breaks the Gherkin document into tokens. The tokens are fed to the next stage.
4. Entity Relationship Extractor: Based on the tokens, it builds a profile of the possible class and method and member names. Then it has to match them with the entities in the Entity Graphs.
5. Test Code Generator: Based on the output of the previous stage, generate code for the target language.
6. GUI: Create new project, open directory, open project, refresh.

Learning to Generate Pseudo-code from SourceCode using Statistical Machine Translation: Generate BDD

Dynamic Test Case Generation using NeuralNetworks by Abhas Kumar

A single step corresponds to one or more code lines. (Snippet)