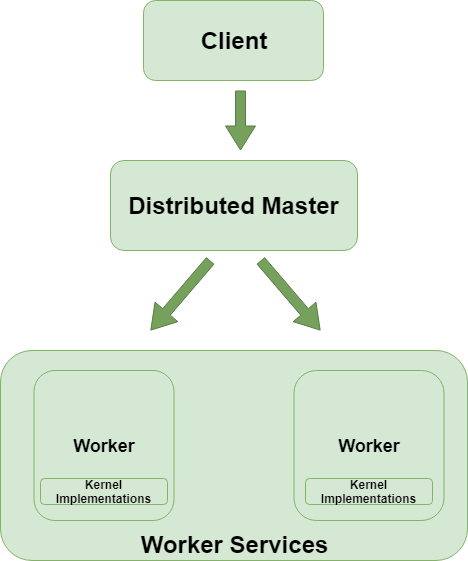
**Case Study: TensorFlow**

TensorFlow is a software application for us to implement neural networks algorithms which is the most important part in machine learning. In this case study, I am going to analyze TensorFlow package through several aspects. They are Technology and Platform used for development, integration tests and the test framework, software architecture and some defects in the project. In the end, I am going to explain my own application to show how TensorFlow is used.

First, Python is the first well-supported language of TensorFlow for expressing and controlling the training of models. However, the core of TensorFlow is not written in Python but in C++ and CUDA because both provide incredible performance in CPU and GPU by using Eigen and NVIDIA’s DNN. As far as I have learnt, I will still choose Python as the front-end coding language of TensorFlow to control and modify the data flow graph because Python is the most comfortable language for us to learn and read compared to other modern languages such as java and C++. With Python, we can express and modify the neural network much easier. Python is easy to integrate, and it has extensive support libraries. As the back-end coding language, C++ is definitely a great choice because a skilled implementation using C++ will have much faster performance than other high-level programing languages. There are several build systems for TensorFlow. According to TensorFlow official website, Bazel is the build tool used to compile TensorFlow. MSYS2 is the bin tools needed to build TensorFlow. Visual Studio 2015 is also needed because it is the environment for the Visual C++ build tools 2015. As I have mentioned, TensorFlow uses Eigen which is a high performance dense linear algebra library. If deep learning algorithm runs in GPU, TensorFlow will use CuDNN which is a very optimized DNN library. Usually, powerful GPU will have greater speed of training than same level CPU.

Second, TensorFlow uses Continuous Integration to ensure that any new changes would not break the whole project. According to TensorFlow GitHub README.md in the ci\_build file folder, Jenkins and CI system internal to Google are the CI platform where they run builds and tests. Inside the ci\_build file folder, there are computing platforms osx(macOS), Windows and Linux to be tested on the CI platforms. When I dig into the Linux file folder, I found out that CMake is the build system it uses. Even though TensorFlow does not list on their README.md, I found out that it also uses Travis-CI as one of its CI platforms through digging the webpage where the stuffs display their code coverage. The web host is called COVERALLS. There is one recent build on branch master on 07 Mar and it got a 81.98 percent code coverage. It does not use any popular code coverage metrics such as branch coverage and sequence point coverage. It only measured relevant lines covered and hits per line. In this case, it got 1492 of 1820 relevant lines covered and 1.64 hits per line.

Third, developers can conveniently use TensorFlow from external projects by builds the computation graph themselves after understanding the basics of convolutional neural network. This is based on the software architecture of TensorFlow that users can define and edit the dataflow graph directly at the Client and the Distributed Master would convert the graph you build to pieces that worker services can understand and do the real work. Therefore, it makes TensorFlow lean more towards functional components. To increase the accuracy of projects, developers could edit the weights and bias or add more layers to the neural network and etc. There are a lot of choices that TensorFlow provides which can make the neural network better at client. TensorFlow simplified architecture is shown below.

TensorFlow architecture consists of four main layers. They are Client, Distributed Master, Worker Services and Kernel Implementations. The Master and Workers layers are the parts where asynchronous progress happens. The Distributed Master distributes the graph pieces to worker services as its name suggests. According to TensorFlow official website, the Worker Services use kernel implementations appropriates to the available CPUs and GPUs to execute the graph operations It also communicate with each other by sending and receiving operation results. Each worker focuses on one task simultaneously so that the whole program runs very fast. TensorFlow architecture is very close to Master-slave pattern which consists of two parties: master and slaves. The master and slaves parties are like the Master and Workers layers here.

In short, Developers edits and defines the flow graph in Client. Client sends a dataflow graph to the Distributed Master. Then, the Master prunes a specific subgraph from the graph and distributes the graph pieces to worker services after partitioning the subgraph into pieces. Finally, workers execute different operations based on kernel implementations and output combined results.