# Dependencies

* For MapReduce
  + multiprocessing
  + zmq
  + os
  + pickle
  + time
* For Testing
  + subprocess
  + socket
  + sys
  + pickle
  + time

# Master Class

The Master class is the primary driver of the distributed process. An instance of the Master class takes the desired IP address, the desired number of mappers and reducers, and port numbers for the mappers and reducers. Once an instance of the master class is created it can call the run\_mapred method. This method takes the input files, the map and reduce functions, and the output location as arguments. The run\_mapred method starts the MapReduce process.

# ProcessInfo Class

ProcessInfo is a class that is used to communicate information between processes and is sent with every message. ProcessInfo holds a process ID, an IP address, a port number, and a logical clock.

# Mapper

Mapper is the process that maps the data given to it using a provided map function. The Master class runs instances of mapper. When mapper finishes processing data it messages that data back to a subscriber set up in Master.

# Reducer

Reducers is the process that reduces the data given to it using a provided reduce function. The Master class runs instances of reducer. When reducer finishes processing data it messages that data back to a subscriber set up in Master.

# Data flow

In Master the text data is read from the provided files and is cleaned. The cleaned data is then chunked into equal portions, one for each mapper. These chunks are provided to the mapper at runtime. The mapper applies the map function to the data and sends it back to Master. In Master the data is sorted. This sorted data is once again chunked into equally sized pieces, one for each reducer. These chunks are provided to the reducer at runtime. The reducer applies the reduce function to the data and sends it back to Master. Master writes this data to the specified output file.

Diagram, engineering drawing

Description automatically generated

# Testing Information

There are two implementations of MapReduce that can be tested, word count and inverted index. These functions are set up in files that can be run with command line arguments. There are three ways to test the implementations. The first option is a user input implementation that allows you to choose between word count and inverted index and set all parameters at run time. The second is the configure.py file. The parameters in that file can be edited and it will run with the selected parameters. The third is that default testing options that I have set up, these will run automatically. These options can be selected out of a menu by running runnable.sh. All output files will be written to the MapReduce/bin directory. Additionally, there are text files in the MapReduce/data directory that can be used for testing. Any other text documents you wish to test should be added to this folder.