**Architecture**

The program consists of two decupled primary modules: application layer and link layer. On the other hand, the link layer module depends on a set of secondary modules: alarm, serial port and utilities.

The link layer is a set of functions that create an API. It’s responsible for developing a protocol that allows it to open/close the serial port connection, read/write data from/to the serial port buffer and recover from errors and interruptions.

The application layer is a generic program that uses the link layer API to send data through a serial port. Its purpose is to read a file’s data and call the link layer functions to open/close the serial port and read/write the data. It’s responsible for adopting an application-specific protocol, so that both sender and receiver work in sync.

**Code Structure**

**Common call stack**

**Primary data structures**

**Primary functions**

**Protocols**

**Link Layer Protocol**

**Application Layer Protocol**

**Usage**

Firstly, execute ‘make’ inside the sources directory to compile the program. Both sender and receiver are the same application, though they receive different command line arguments. To launch the application, type the name of the executable (i.e. ‘./app’). Then, you should type the port (e.g. /dev/ttyS0) and the mode on which the program should operate (i.e. ‘receiver’ or ‘sender’).

The sender program will also need to receive as a command line argument the path of the file to send (e.g. ‘banana.gif’). Optionally, you can specify the amount of file data in bytes to send per packet (e.g. ‘1024’).

Example of usage:

* ‘./app /dev/ttyS0 receiver’
* ‘./app /dev/ttyS0 sender banana.gif 1024’