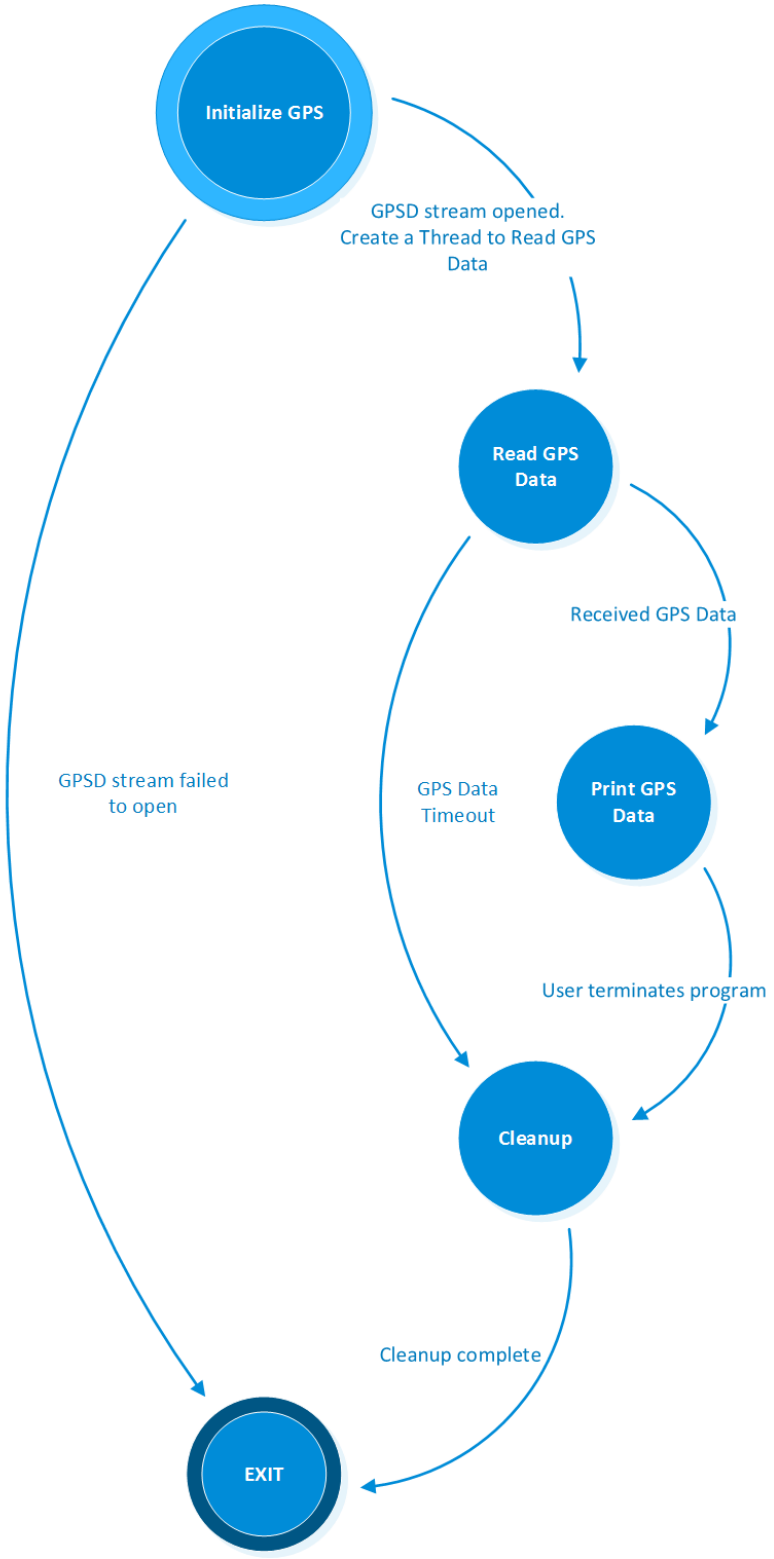
DESIGN & IMPLEMENTATION

GPS APPLICATION

Table of Contents

1 State Diagram



2 State Implementation

State	Details
Initialize GPS	*Purpose*: Starting point for the user into the program. Once the program is started, the following events take place: • Reserve memory (malloc) for the gps dasta structure that holds all the gps data that will be retrieved from the gps dongle. • Set the data source information; in this case it would be the localhost, since we are accessing the gps dongle plugged in via USB • Try to open up the socket to the gpsd daemon. The session socket needs to be successful if we are to retrieve gps information. • Once the socket is successful, we need to issue a WATCH command to tell the daemon to start streaming GPS data to our application. • Once all the above is successful, we create a thread to read GPS data, and pass in our gps data structure. • We also define and create all the necessary front end windows to display our GPS data later on. **GPSD functions used:** **gps_open** **gps_stream**
Read GPS Data	*Purpose*: This will be run as a separate thread, which is created when we Initialize the GPS. The purpose here is to check if there is some GPS data to be read from the daemon. If there is, we simply write the GPS data into the structure we defined earlier, and pass this structure to the Print function to display the data. When we can't get GPS data or when the gpsd times out, we issue a cleanup call on the gpsd stream. **GPSD functions used**: **gps_waiting** **gps_read**
Print GPS Data	*Purpose:* Print the GPS data that is passed in via a structure from the Read GPS loop. The following is printed to the screen: Time(UTC), Latitude, Longitude, Speed, Status (No Fix/2D/3D), PRN, Elevation, Azimuth, SNR, and Whether the Satellite is being used.
Cleanup	*Purpose:* Disables the WATCH command for the gpsd stream so it will stop sending data, close the gpsd daemon socket, and free the memory reserved for the gps data structure. **GPSD functions used:** **gps_stream** **gps_close**

3 Pseudocode

Initialize GPS
	Malloc gps data structure to hold gps data
	Setup data source information with port and address.
	Attempt to open up a session socket to the daemon.
		If we get a handle on the GPS dongle, issue a WATCH command to get gps data stream.
		If opening gpsd fails, print error and exit.
	Create display UI to display gps information.
	Once session socket acquired and gpsd stream open, create new thread to **Read GPS Data.**

Read GPS Data (Run in it's own thread)
Loop
	Check to see if there is gps data waiting from the daemon.
		If there is, read and write it into the gps data structure.
			Print GPS Data to the screen.
		If there isn't, call **Cleanup**.
	If GPS Data times out, call **Cleanup**.
End Loop

Print GPS Data
	Use the GPS data structure passed in to access data to be printed.
	Print the following data:

Cleanup
	Disable the WATCH command on the gpsd stream.
	Close the gpsd stream and socket.
	Free the malloc'ed structures.