Libreria utilizzata: PracticalSocket.h

```
Spykee::Spykee(char* user, char* pass)
Spykee::~Spykee()
void Spykee::unplug()
void Spykee::setplug()
void Spykee::soundAllarm()
void Spykee::soundBomb()
void Spykee::turnOnCameraLight()
void Spykee::turnOffCameraLight()
void Spykee::move(int leftSpeed, int rightSpeed)
void Spykee::startCamera()
int Spykee::getState()
static unsigned char* pharseMessage(unsigned char* m, int* lm, unsigned char **
image, enum operations op)

enum operations
{
      pharseNewMessage,
      takeChargeState,
      takeCameraPictureWithLock,
      unlockPicture
}

enum states
{
      Connected = 1,
      Moving = 2,
      CameraOn = 4
}
```

Spykee::Spykee(char* user, char* pass)
Costruttore della classe Spykee. Cerca uno Spykee connesso alla rete, si connette e ne esegue l'autenticazione con user e password passati come parametri (di default user: admin, password: admin).

Spykee::~Spykee()
Distruttore della classe Spykee.

void Spykee::unplug()

Metodo che invia il comando a Spykee per farlo staccare dal caricabatterie.

void Spykee::setplug()

Metodo che invia il comando a Spykee per farlo attaccare al caricabatterie.

void Spykee::soundAllarm()

Metodo che invia il comando a Spykee per fargli eseguire un suono tipo allarme.

void Spykee::soundBomb()

Metodo che invia il comando a Spykee per fargli eseguire un suono tipo bombe.

void Spykee::turnOnCameraLight()

Metodo che invia il comando a Spykee per fargli accendere la luce sotto la telecamera.

void Spykee::turnOffCameraLight()

Metodo che invia il comando a Spykee per fargli spegnere la luce sotto la telecamera.

```
void Spykee::move(int leftSpeed, int rightSpeed)
```

Metodo che invia il comando a Spykee per farlo muovere. Le velocità vanno da 90 a -90. Velocità positive fanno muovere i cingoli in avanti, velocità negative fanno muovere i cingoli in dietro.
leftSpeed indica la velocità del cingolo sinistro.
rightSpeed indica la velocità del cingolo destro.
Per far ruotare il robot basta semplicemente muovere un cingolo in avanti e l'altro in dietro.

```
void Spykee::startCamera()
```

Metodo che invia il comando a Spykee per iniziare ad acquisire e inviare immagini.

```
int Spykee::getState()
```
Metodo che ritorna lo stato della connessione a Spykee. La gestione dello stato non è ancora stata completamente implementata.

```
static unsigned char* pharseMessage(unsigned char* m, int* lm, unsigned char * image, enum operations op)
```

Pharsemessage è il metodo che analizza i messaggi arrivati.
Le immagini che arrivano le salva in modo da poterle poi recuperare. Quando arriva un'immagine la salva, appena ne arriva un'altra la salva anch'essa senza però eliminare la precedente. Appena ne arriva un'altra la sostituisce alla più vecchia. In questo modo appena arriva la richiesta di voler prendere un'immagine c'è né subito una pronta.
Per acquisire un'immagine basta richiamare la funzione pharseMessage in questo modo:

```
pharseMessage(void, lm, image, Spykee::takeCameraPictureWithLock)
```

dove lm è un puntatore ad intero e image è un puntatore ad un'array di unsigned char. La funzione ritornerà il puntatore all'ultima immagine ricevuta completamente e lo metterà anche in immage e in lm metterà la lunghezza dell'immagine. L'immagine ritornata rispetterà lo standard JFIF. Appena finito di utilizzare l'immagine bisognerà comunicarlo alla funzione in questo modo:

```
pharseMessage(void, lm, image, Spykee::unlockPicture)
```

e a questo punto sarà possibile richiedere una nuova immagine.

Spykee permette molte altre funzioni che per ora nella classe non sono state implementate, come ad esempio il poter riprodurre qualsiasi suono, l'utilizzo del microfono ambientale, l'utilizzo delle lucine colorate...
Per poter utilizzare ciò è possibile mandare i comandi al robot tramite il software di controllo e nel frattempo catturare i pacchetti che vengono inviati al robot ed interpretarli.