Progetto dighe

AS IS

Sono presenti N di dighe distribuite in tutto il nord Italia. Ogni diga è monitorata localmente tramite sensori, ed è presidiata fisicamente da addetti. Quando l'acqua raggiunge la soglia massima, o si avvicina ad essa, il sistema deve entrare rispettivamente in stato di allerta o stato di preallerta. Lo stato della diga deve essere costantemente monitorato ed aggiornato. Non è garantita in ogni diga la connettività internet ma la connessione GSM.

TO BE

Obiettivo del progetto: rimozione del presidio fisico e sostituzione con un nuovo sistema di monitoraggio automatico Il nuovo sistema di monitoraggio automatico prevede: per ogni diga, un computer, collegato ad un router GSM, che riceve dati dal sensore; nella sede centrale di Piacenza, un computer, collegato ad un router GSM, che in tempo reale riceve lo stato dell'acqua presente in ogni diga.

Funzionamento del sistema: quando l'acqua raggiunge lo stato di preallerta o allerta, il computer situato sulla diga, tramite router GSM, invia al computer della sede centrale un SMS contenente il rispettivo stato; il computer centrale riceve in tempo reale gli stati delle dighe e aggiorna continuamente la sua interfaccia; periodicamente, ogni computer di diga manda al server centrale un messaggio di keep-alive, in modo da assicurarsi che il collegamento funzioni e sia attivo. Devono essere presenti gruppi di continuità in ogni diga (UPS).

Analisi progetto

Il nostro obiettivo è **rimuovere il presidio fisico** e **utilizzare dei sistemi automatici**. Si ha questo bisogno perché togliendo l'uomo si riducono i costi (3 persone = 3 turni), si evita l'errore umano, si ha una raccolta dati efficiente. Ovviamente ci sarà una squadra di manutenzione che controllerà il funzionamento dell'infrastruttura.

I **sensori** sono già presenti, saranno sempre attivi e dovranno comunicare lo stato anche quando andrà bene. Questi sensori manderanno informazioni al computer che tramite il router GSM manderà in tempo reale un SMS.

I sensori individueranno tre stati:

- **Nessuna allerta**
- **Preallerta**
- **Allerta**

Per identificare il messaggio che verrà mandato dal Router della diga al Router centrale il **pacchetto** sarà composto nel seguente modo: "**%**" per identificare il messaggio, **il codice per l'identificazione della diga** (sarà di due caratteri), l'identificazione può essere anche identificata dal numero (verrà fatto un controllo che il numero telefonico corrisponda al numero identificativo), e infine lo **stato dell'acqua** (1 carattere, 1 = preallerta, 2 = allerta, quando non c'è

nessuna allerta non viene mandato nessun messaggio) oppure "**ka**" per identificare il messaggio di keep-alive. Quindi i pacchetti potranno essere "*%011*" o "*%01ka*".

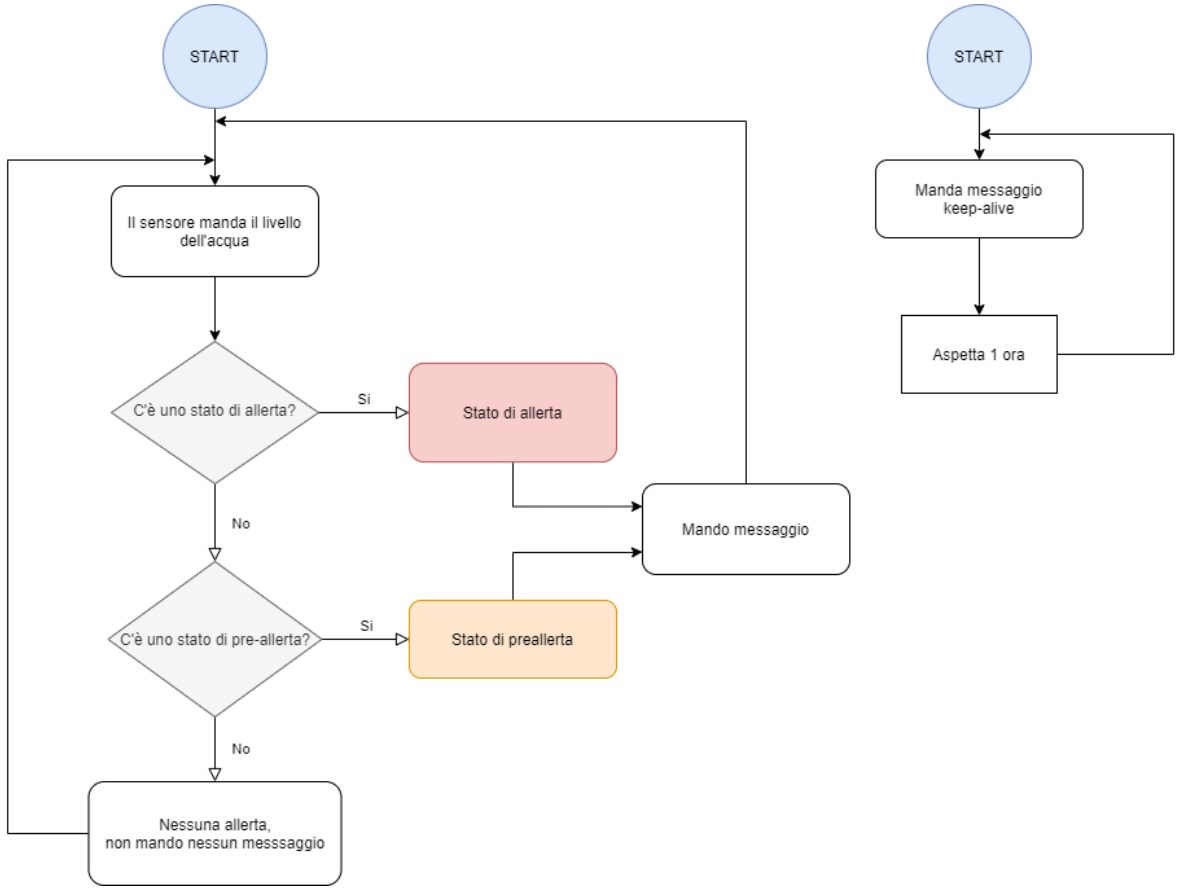
Ci sarà un **sinottico** che mi farà vedere lo stato delle dighe.

Keep-alive: mandiamo un messaggio al server centrale che assicura il corretto funzionamento. È fondamentale per controllare che la diga sia online, in caso contrario bisogna mandare degli operatori ad intervenire. In questa situazione mandiamo il messaggio keep-alive **una volta all'ora**.

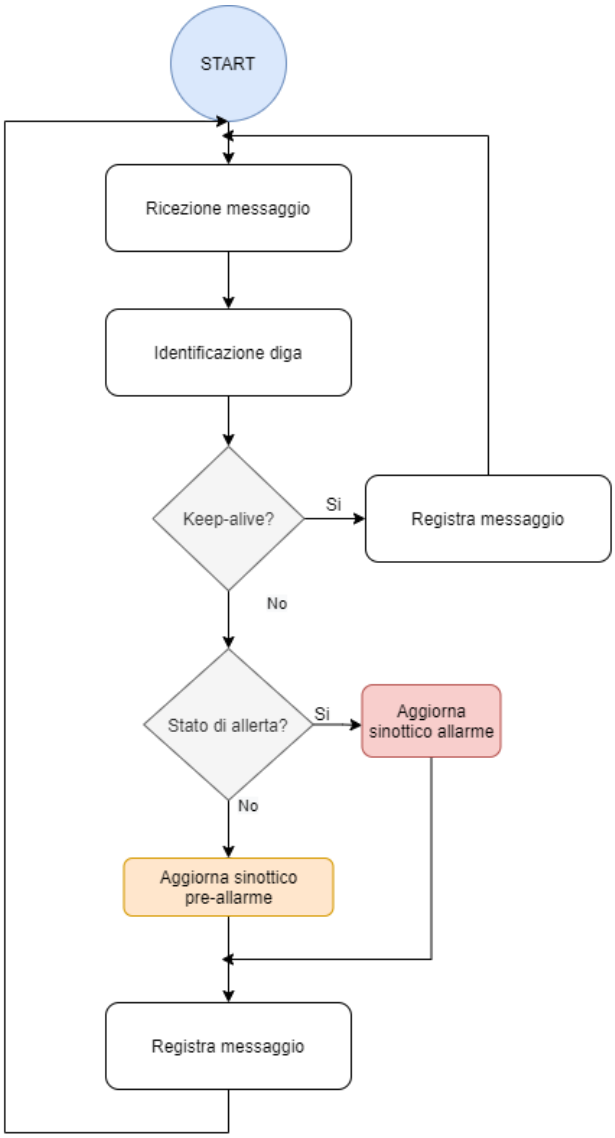
Non usiamo lo stato di no allarme per evitare di mandare informazioni inutili. Non dobbiamo segnalare che tutto vada bene, dobbiamo **segnalare l'anomalia**.

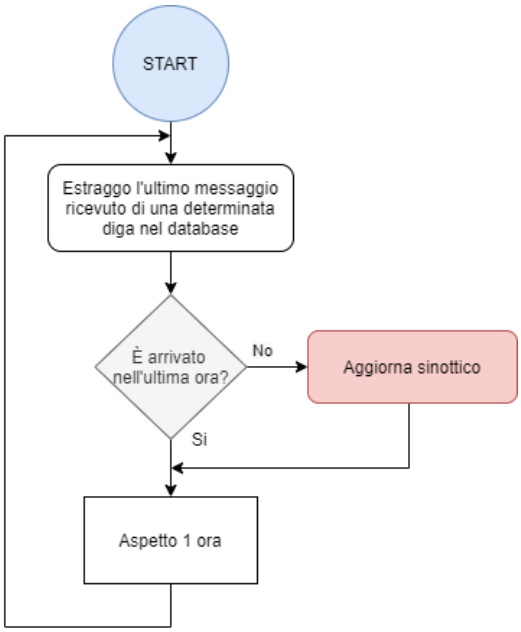
Schema logico

PC diga

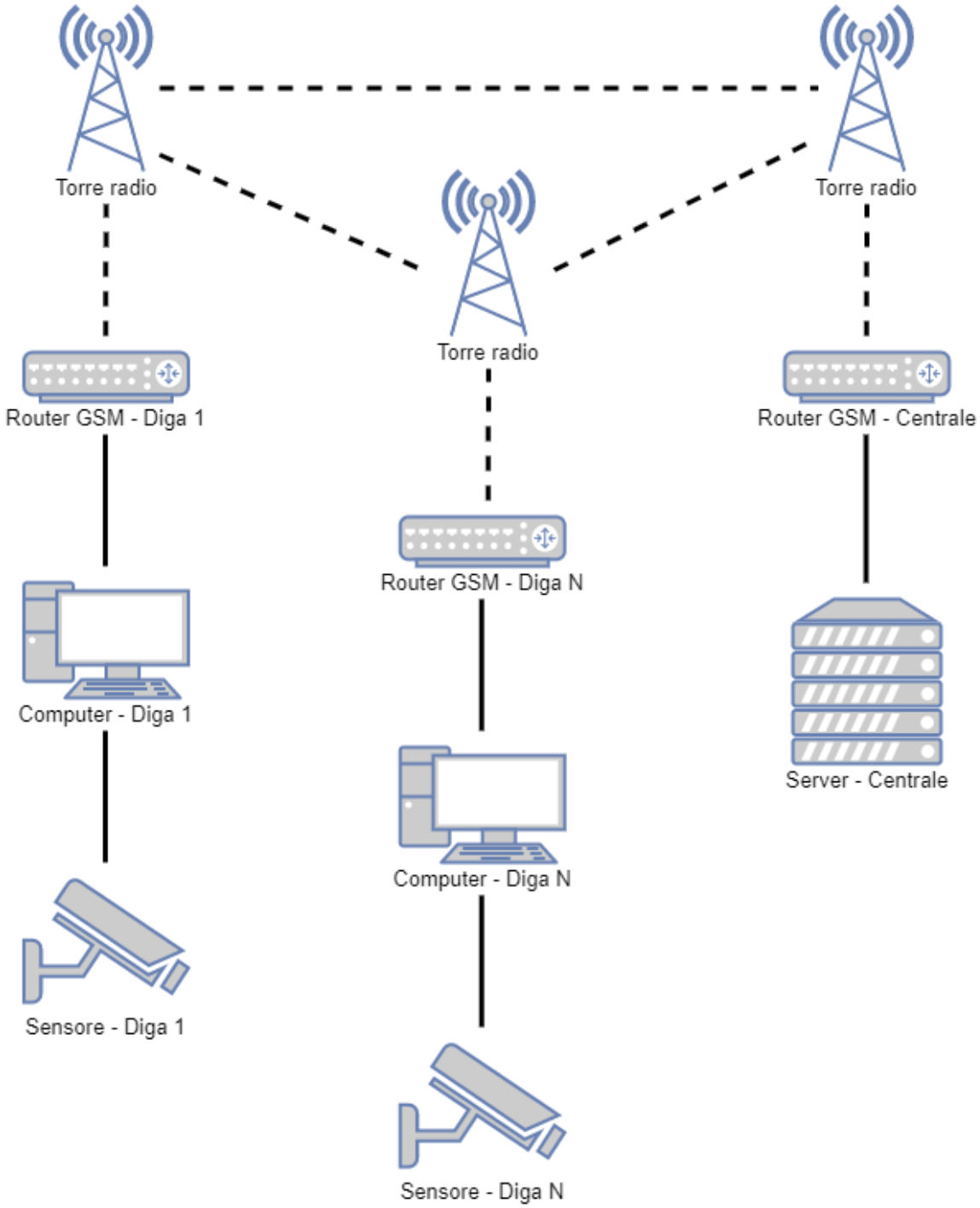


PC centrale

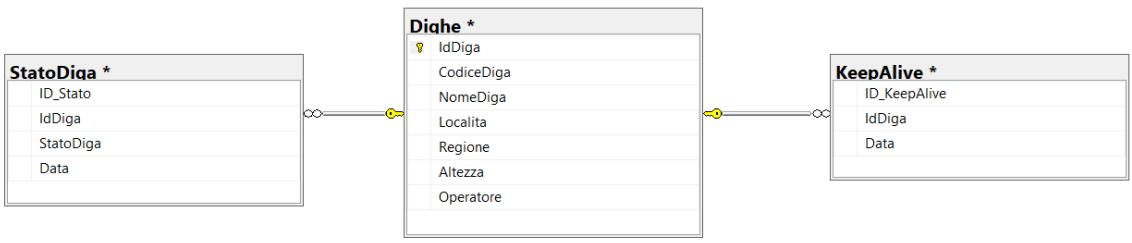

START
Ricezione messaggio
Identificazione diga
Keep-alive?
Si
Registra messaggio
No
Stato di allerta?
Si
Aggiorna sinottico allarme
No
Aggiorna sinottico pre-allarme
Registra messaggio


START
Estraggo l'ultimo messaggio ricevuto di una determinata diga nel database
È arrivato nell'ultima ora?
No
Aggiorna sinottico
Si
Aspetto 1 ora

Schema fisico



Struttura database



Codice

PC diga

Questo processo si occuperà di **identificare** se il segnale ricevuto dai sensori è preallerta o allerta, se è così allora **manderà il messaggio** attraverso il router GSM al PC centrale. Se non c'è nessuna allerta non dobbiamo segnalarlo, sarebbe solo uno spreco di risorse.

Come ho spiegato nell'analisi del progetto il nostro pacchetto sarà costruito nel seguente maniera: **"%" + codice diga + stato diga**. Il percento serve per identificare il messaggio, quando il router del server centrale legge quello che c'è nel buffer legge anche tutti i comandi AT e le date quindi abbiamo bisogno di identificare il nostro messaggio.

```
srl_routerGSM.Open();

// Preallerta = 1
// Allerta = 2
// Esempio "%011"
string messaggio = "%" + codDiga + statoDiga;

// Comandi per mandare il messaggio attraverso il router GSM
srl_routerGSM.WriteLine(@"AT+CMGF=1" + (char)(13));
Thread.Sleep(2000);

// Qui andrà il numero del PC centrale
srl_routerGSM.WriteLine(@"AT+CMGS=" + numero + (char)(13));
Thread.Sleep(2000);

// Qui andrà il messaggio che abbiamo composto prima
srl_routerGSM.WriteLine(messaggio + (char)(26));

// Chiudo la porta, per permettere ad altri processi (keep-alive) di utilizzarla
srl_routerGSM.Close();
```

PC diga - keep alive

Questo processo è fondamentale per assicurare che la diga funzioni nel modo corretto.

In questo processo c'è un **timer** che ogni ora manderà il messaggio di keep-alive.

La struttura del nostro messaggio sarà **"%" + codice diga + "ka"**. Il "%" serve per identificare il nostro messaggio, ka sta per keep-alive in questo modo il server potrà riconoscere che il messaggio che gli stiamo mandando è un keep-alive.

Il messaggio verrà mandato nello stesso modo in cui è stato spiegato prima.

```
// Esempio "%01ka"
string messaggio = "%" + codDiga + "ka";
```

PC centrale

Questo programma sarà collegato a un router GSM dal quale leggerà i messaggi, li identificherà, aggiornerà il sinottico e li salverà nel database.

Per leggere i messaggi ci sarà un timer che ogni minuto andrà ad eseguire i seguenti comandi per andare a leggere i messaggi presenti nel buffer:

```
// IMPORTANTE: il buffer deve essere vuoto, questo comando cancella solo l'ultimo messaggio (nella posizione 1) in questo modo andremo a leggere l'ultimo messaggio ricevuto
// Svuoto il buffer
srl_routerGSM.WriteLine(@"AT+CMGD=1" + (char)(13));
Thread.Sleep(2000);

srl_routerGSM.WriteLine(@"AT+CMGF=1" + (char)(13));
Thread.Sleep(2000);

// Leggo tutti gli SMS in memoria, in questo caso ne avremo solo 1
srl_routerGSM.WriteLine(@"AT+CMGL=ALL" + (char)(13));
Thread.Sleep(2000);
```

Una volta eseguiti questi comandi attiveremo il DataReceived della porta seriale e andremo a **leggere il messaggio nel buffer**:

```
private void srl_routerGSM_DataReceived(object sender,
System.IO.Ports.SerialDataReceivedEventArgs e)
{
    //Leggo il messaggio dal buffer
    string messaggioRicevuto = srl_routerGSM.ReadExisting();
}
```

Una letto il messaggio dovremo identificarlo, il problema è che vengono letti anche tutti i comandi e altre informazioni inutili:

```
OK
REC READ","+393000000000",,"21/02/14,13:47:56+04"
%011

+CMGL: 2,"L=ALL
AT+CMGAT+CMGD=1

OK
AT+CMGF=1

OK
```

Ed è per questo il motivo per cui ho aggiunto il carattere "%" all'inizio del messaggio, con questo carattere risulta più semplice selezionare solo il messaggio che mi serve.

Dopo aver fatto un semplice parser e aver estratto il messaggio dobbiamo **identificare** la diga e che tipo di messaggio abbiamo ricevuto. Sappiamo che il nostro messaggio può essere in questi due modi: *"%"* + *codiceDiga* + *statoDiga* ("%011") o *"%"* + *codiceDiga* + *"ka"* ("%01ka").

Con un semplice controllo possiamo capire se si tratta di un messaggio keep-alive o di un messaggio che riguarda lo stato della diga.

Se è un **keep-alive** dobbiamo registrare il messaggio nel database, questa è la query:

```
INSERT INTO KeepAlive(IdDiga, Data)
SELECT IdDiga, GETDATE() FROM dbo.Dighe
WHERE CodiceDiga = @codiceDiga
```

Se invece è un messaggio che serve per lo **stato della diga** dobbiamo capire se è uno stato di allarme o preallarme e quindi aggiornare il sinottico e registrare il messaggio:

```
INSERT INTO StatoDiga(IdDiga, StatoDiga, Data)
SELECT IdDiga, @statoDiga, GETDATE() FROM dbo.Dighe
WHERE CodiceDiga = @codiceDiga
```

L'accesso al database viene effettuato con ADO.NET

PC centrale - keep alive

Questo processo controlla che tutte le dighe mandino regolarmente il **keep-alive**. Ricordo che il keep-alive garantisce il corretto funzionamento del sistema quindi è molto importante anche questo controllo.

La prima cosa che viene fatta è eseguire una query che mi prenda dal database tutti i codici diga in questo modo, avendo un vettore contenente questi codici, potrò controllare la presenza del keep-alive per ogni diga.

C'è un timer che ogni 65 minuti (se il keep-alive viene mandato ogni ora questo controllo lo potremo fare ogni 65/70 minuti, la gestione di questi tempi sarà possibile farlo da interfaccia grafica che andrò a illustrare successivamente) andrà a controllare se negli ultimi 60 minuti è stato registrato un messaggio di keep-alive.

Quindi verrà eseguita questa query per ogni diga:

```
/*Viene preso l'ultimo keep-alive e poi viene fatto il controllo*/
SELECT TOP 1 Dighe.CodiceDiga, Data FROM KeepAlive
INNER JOIN dbo.Dighe ON KeepAlive.IdDiga = Dighe.IdDiga
WHERE KeepAlive.IdDiga = @IdDiga
ORDER BY Data DESC
```

Una volta eseguita la query verrà controllato se effettivamente l'ultimo keep-alive è arrivato nell'ultima ora:

```
SqlDataReader reader = cmd.ExecuteReader(CommandBehavior.CloseConnection);
while (reader.Read())
{
    DateTime data = DateTime.Parse(reader["Data"].ToString());
    // Faccio la differenza tra la data attuale e la data dell'ultimo keep alive
ricevuto
    TimeSpan diff = DateTime.Now.Subtract(data);
    double diffMinuti = Convert.ToDouble(diff.TotalMinutes);

    // Se sono passati più di 60 minuti dall'ultimo keep alive segnalo
```

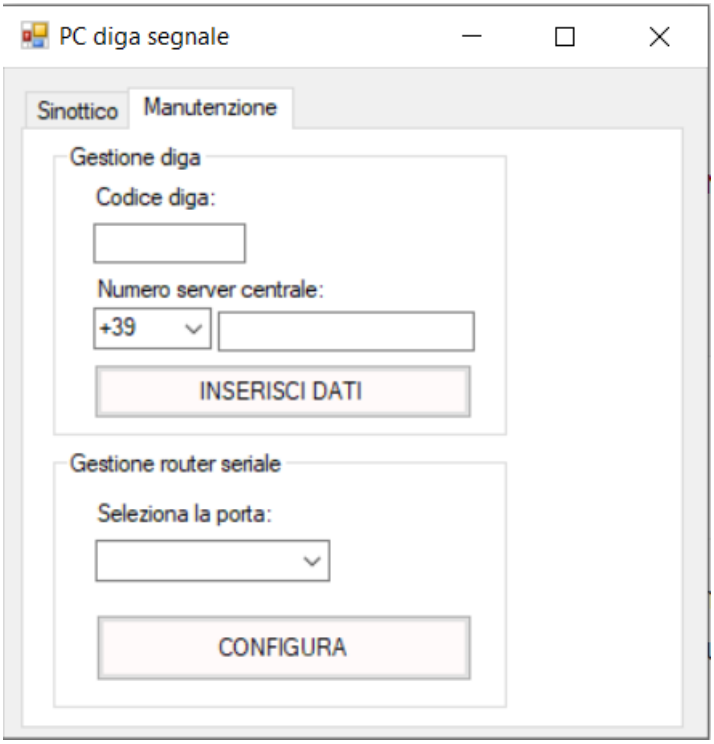
```
    if (diffMinuti > minuti)
        {
            // Aggiorno sinottico
            ListViewItem lvItem = new ListViewItem(new string[]
                                                    {
reader["CodiceDiga"].ToString(), "NO KEEP-ALIVE" });
            lv_sinottico.Items.Add(lvItem);
        }
}
```

Manuale d'uso

Ogni programma ha un'interfaccia destinata sia all'utente finale che al tecnico che dovrà andare a sistemare qualche malfunzionamento che. oltre i classici metodi di manutenzione (spegni e riaccendi) potrà modificare alcune "impostazioni" del programma.

PC diga

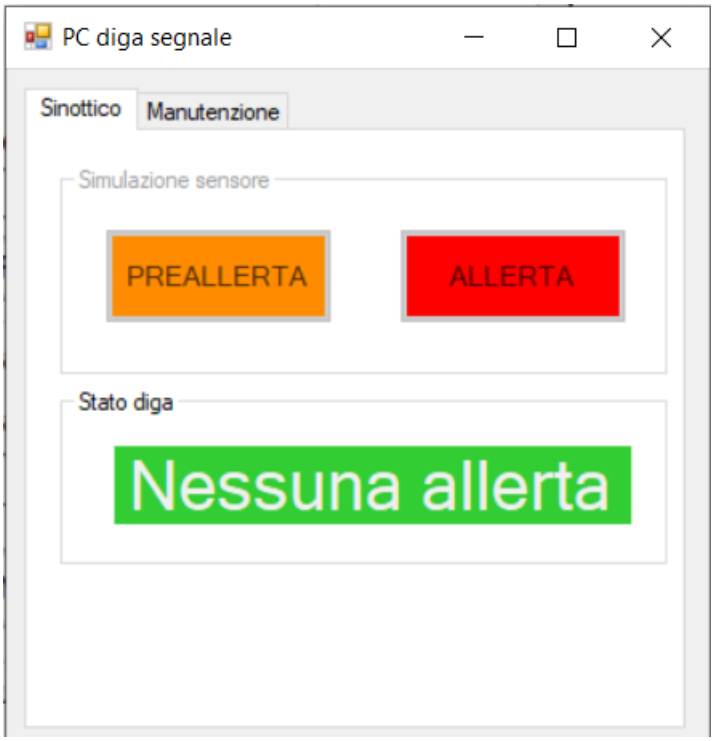
All'avvio di questo programma questa sarà la prima interfaccia che ci verrà proposta:



Come prima cosa si dovrà inserire il codice identificativo della diga, che è di due caratteri (es: "01"), poi il numero di telefono del pc centrale. Una volta inseriti questi dati bisognerà premere il bottone "Inserisci dati".

Nella seconda parte di questa tab bisognerà selezionare nella combobox la porta seriale del router GSM.

Successivamente ci si dovrà spostare sulla tab "Sinottico":



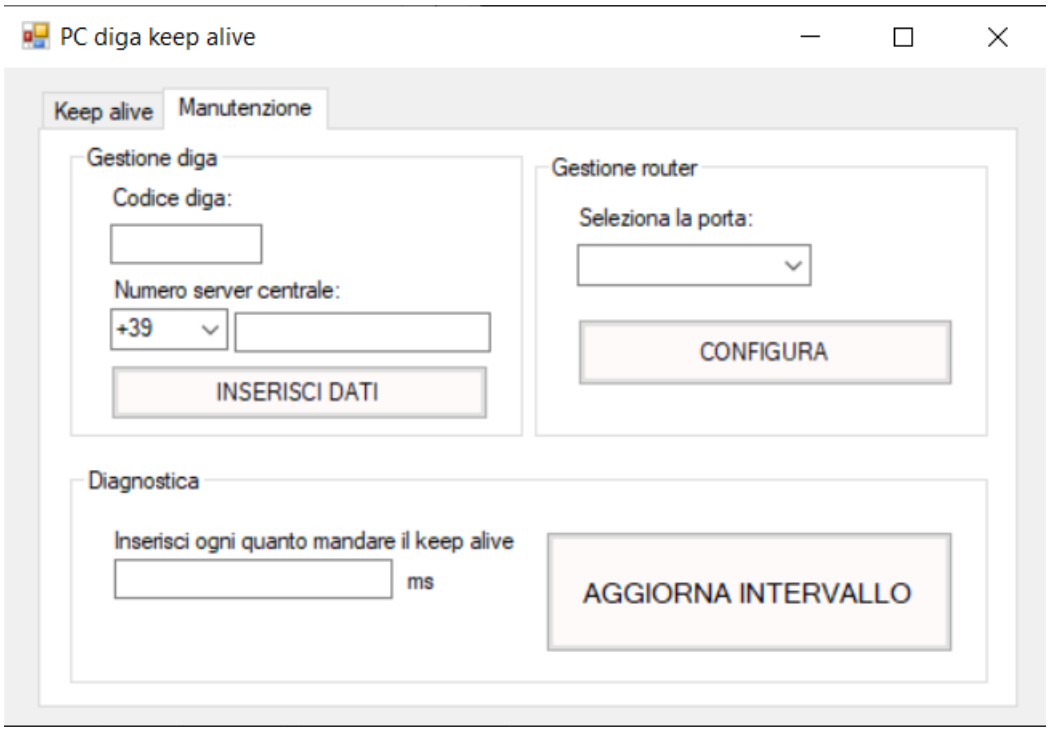
Dato che nella fase di realizzazione di questo progetto non ho avuto a disposizione l'arduino per la ricezione dei segnali l'ho dovuto simulare con due bottoni. Premendo uno dei due tasti manderemo il messaggio al router GSM del PC centrale.

Sotto c'è un semplice sinottico dove verrà aggiornato lo stato della diga in modo automatico.

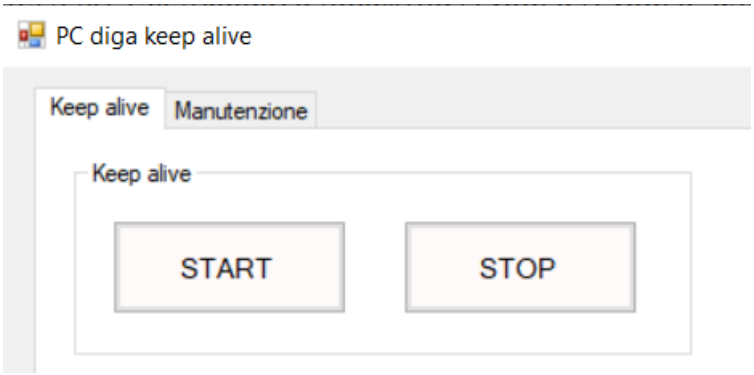
PC diga keep alive

Le sezioni "Gestione diga" e "Gestione router" vanno compilate come è stato spiegato prima.

Nella sezione "Diagnostica" verrà inserito,in millisecondi, ogni quanto mandare il messaggio di keep-alive, una volta inserito il tempo bisogna premere il tasto "Aggiorna intervallo". Questo dato è molto importante, infatti, dovrà stabilirsi uno standard per tutte le dighe, quello scelto da me è 60 minuti, un tempo errato andrà a falsare il controllo che avviene nel lato PC centrale. Questo sezione è stata progettata per permettere al tecnico di "debuggare" il programma senza mettere mano al codice.

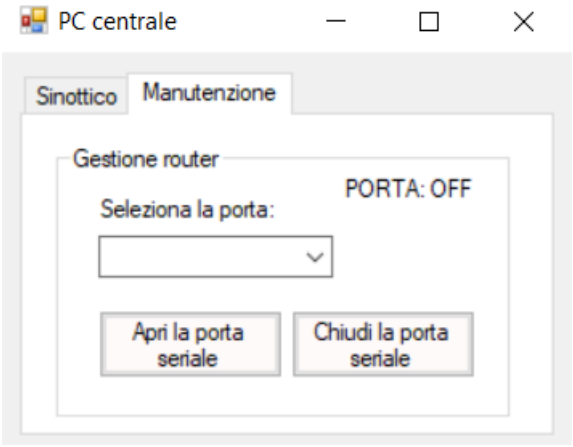


Nella tab "Keep-alive" ci sono due semplici bottoni per far partire e fermare il timer:

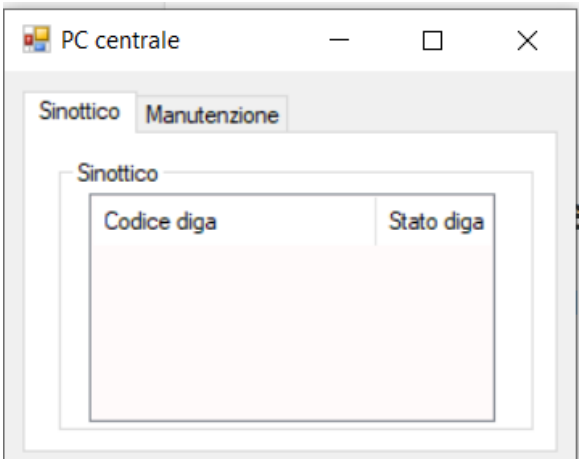


PC centrale

Qui molto semplicemente bisogna scegliere la porta seriale del router GSM e aprirla

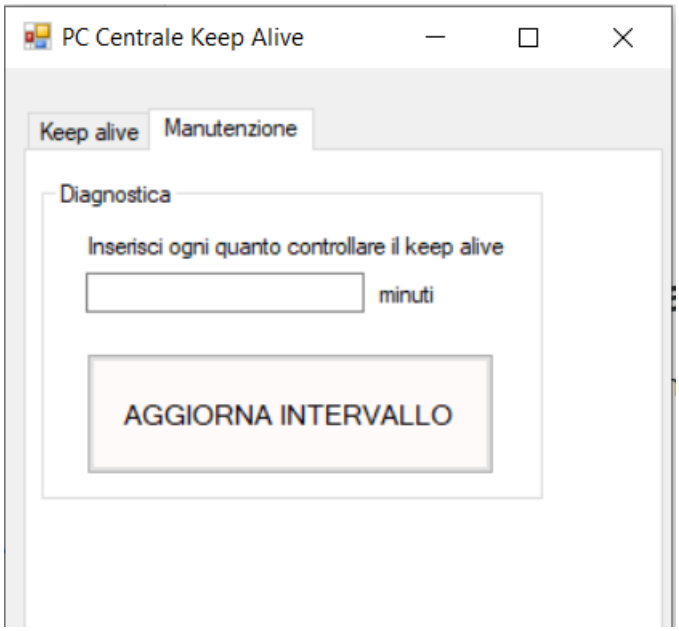


Nella tab "Sinottico" avremo una tabella che verrà aggiornata ogni volta che che una diga avrà un segnale di allarme o di preallarme.



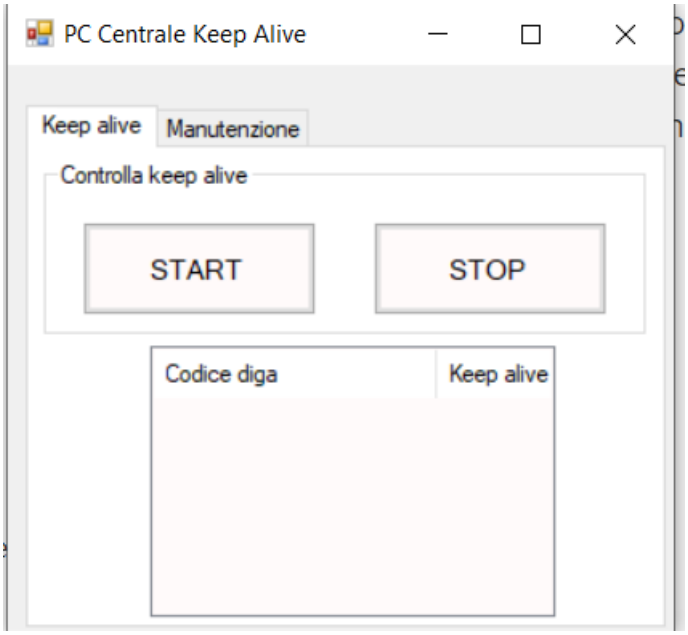
PC centrale - keep alive

In questa sezione andrà inserito in minuti ogni quanto si andrà a controllare il funzionamento del keep alive. Questo dato va' inserito in base allo "standard" deciso per le dighe. Nel mio caso il keep alive verrà mandato ogni 60 minuti, mentre il controllo verrà fatto ogni 65 minuti.



Nella tab "Keep alive" dovrò far partire il timer attraverso il tasto "Start" e potrò stopparlo in caso di necessità.

Sotto ho una tabella che verrà aggiornata nel momento in cui un PC diga smetterà di mandare regolarmente il keep alive.



Autore

Kristian Kovacev - 5F