

SAMUELE MOSCATELLI, NICOLÒ PINCIROLI,
ANDREA POZZOLI

GESTIONE DEL TRAFFICO



DOCUMENTAZIONE DEL PROGETTO DI
PROVA FINALE

POLIMI

INGEGNERIA INFORMATICA

A.A. 2018 - 2019

Samuele Moscatelli, Nicolò Pincioli,
Andrea Pozzoli,
Gestione del traffico
Documentazione del progetto di prova finale

E-MAIL:
Samuele Moscatelli - sem.mosca97@libero.it
Nicolò Pincioli - nicolopinci.1997@gmail.com
Andrea Pozzoli - pozzoliandrea97@gmail.com



Figura 1: Un esempio di traffico

INDICE

1	IMPLEMENTAZIONE IN JAVA	1
1.1	Algoritmi significativi	1
1.1.1	Calcolo dell'intervallo di aggiornamento della centralina	1
1.1.2	Variazione di velocità rilevata dalla centralina	2
1.2	Interfaccia grafica	2
1.2.1	Sistema centrale	3
1.2.2	Centralina stradale	3
1.2.3	Applicazione mobile	4
1.2.4	Utilizzo di Swing	4
1.2.5	Login e registrazione	5
1.3	Formato delle notifiche	5
1.4	Casi di test JUnit	6
1.5	Sintesi	6
BIBLIOGRAFIA		7

1

IMPLEMENTAZIONE IN JAVA

Il progetto complessivo è costituito da tre progetti, collegati tramite RMI. I progetti realizzati sono **Sistema centrale**, **Centralina¹** e **Applicazione mobile**.

1.1 ALGORITMI SIGNIFICATIVI

1.1.1 Calcolo dell'intervallo di aggiornamento della centralina

```
float temp=((float)numeroVeicoli)/((float)this.  
    intervalloDiTempo);  
  
if (temp>0.0) {  
    this.intervalloDiTempo=(int) (this.intervalloDiTempo  
        *this.rapporto/temp);  
  
    if (this.intervalloDiTempo<this.intervalloMinimo) {  
        this.intervalloDiTempo=this.intervalloMinimo  
        ;  
    }  
  
    this.rapporto=temp;  
}  
else {  
    this.intervalloDiTempo=this.intervalloDiTempo*2;  
}
```

L'algoritmo, come da specifica, varia l'intervallo di tempo di segnalazione del traffico da parte della centralina in base alla quantità di veicoli rilevata. In particolare, se vengono rilevati veicoli, `temp` sarà maggiore di 0, di conseguenza il nuovo intervallo di tempo dipenderà dall'intervallo di tempo precedente (impostato dall'interfaccia della centraline), dal rapporto (inizialmente a zero) e da `temp` stesso.

L'intervallo di tempo ha un valore minimo, sotto il quale non è possibile scendere. Se non vengono rilevati veicoli, invece, viene raddoppiato l'intervallo di tempo considerato.

All'aumentare del numero di veicoli per unità di tempo rilevati, quindi, diminuisce l'intervallo di tempo della rilevazione successiva.

¹ È stata implementata la centralina stradale

1.1.2 Variazione di velocità rilevata dalla centralina

```
Random random = new Random();
int a;
if (this.velocita>15) {
    a = this.velocita-15;
}
else {
    a = 0;
}
int b = this.velocita+15;
int c = ((b-a) + 1);

int vel = random.nextInt(c) + a;
this.sommaVelocita=this.sommaVelocita+vel;
```

Una centralina stradale deve rilevare la velocità dei veicoli che transitano sul tratto di strada a cui si riferisce.

Dal momento che risulterebbe poco realistico se la centralina rilevasse una velocità molto diversa da quella rilevata durante la rilevazione precedente, la simulazione impone che questa possa aumentare o diminuire fino a 15 km/h da una rilevazione a quella successiva.

Nel caso in cui la velocità dovesse essere inferiore a 15 km/h, il valore minimo consentito per la rilevazione successiva sarà pari a 0 km/h².

Il valore minimo, il valore massimo e il range tra i valori minimo e massimo vengono indicati dalle variabili a, b e c.

Una volta generata la velocità casuale, questa viene aggiunta a *sommaVelocita*, che viene utilizzata per il calcolo della velocità media (infatti viene divisa per il numero di automobili rilevate).

1.2 INTERFACCIA GRAFICA

L’interfaccia grafica è stata realizzata per agevolare l’interazione con gli utenti e con gli amministratori. In particolare, per ogni progetto (centralina stradale, sistema centrale e applicazione mobile) è stato realizzata un’interfaccia diversa.

L’interfaccia grafica è stata realizzata utilizzando Swing [VVe] ed è stata integrata con le API fornite da OpenStreetMap [VVb], distribuito con licenza GPL.

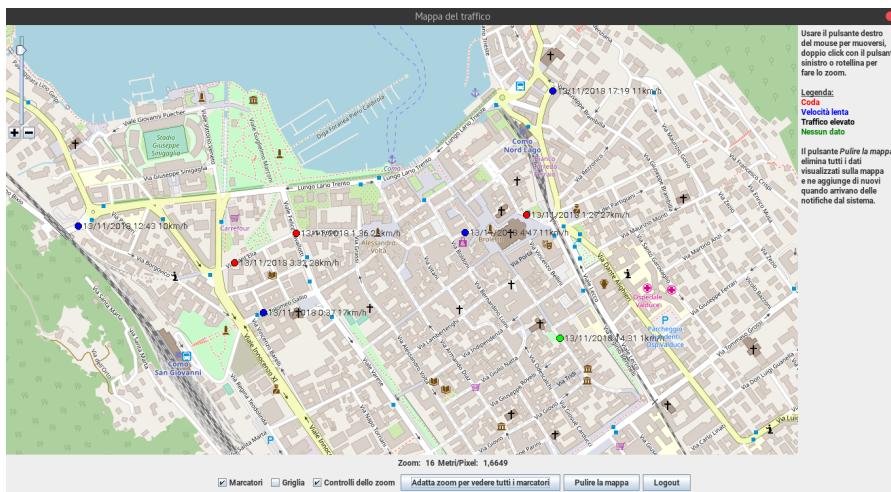


Figura 2: La mappa con alcuni marcatori posizionati casualmente

1.2.1 Sistema centrale

L’accesso al sistema centrale viene permesso in seguito al **login** o alla **registrazione** di un nuovo amministratore. La prima schermata visualizzata, quindi, richiede di effettuare una di queste due operazioni.

Se il login o la registrazione vanno a buon fine viene visualizzata l’interfaccia grafica vera e propria del sistema centrale, costituita dalla mappa (i dati provengono da [Vc]), da alcuni controlli per spostarsi sulla mappa e da un pulsante che permette di effettuare il logout dal sistema centrale.

Quando la mappa riceve dei dati dagli altri sistemi (collegati grazie a RMI[Vd]), li visualizza utilizzando marcatori di colore diverso in base all’evento di traffico (la legenda³ sulla destra dell’interfaccia grafica spiega il significato dei diversi colori), riportando, per ogni marcitore, informazioni relative alla data e all’ora di rilevamento e alla velocità rilevata (se disponibile). I marcatori vengono automaticamente posizionati in corrispondenza della posizione della segnalazione.

Nel caso in cui più marcatori si riferiscano alla stessa posizione viene mantenuta l’ultima informazione ricevuta dal database e i marcatori già presenti vengono eliminati dalla mappa.

1.2.2 Centralina stradale

La centralina stradale viene impostata senza effettuare il login e la registrazione, infatti immaginando che il sistema venga effettivamente realizzato è ragionevole pensare che la centralina venga impostata,

² Una velocità negativa non sarebbe realistica nel sistema reale

³ **Rosso:** coda; **Nero:** traffico elevato; **Blu:** velocità lenta; **Verde:** altro.

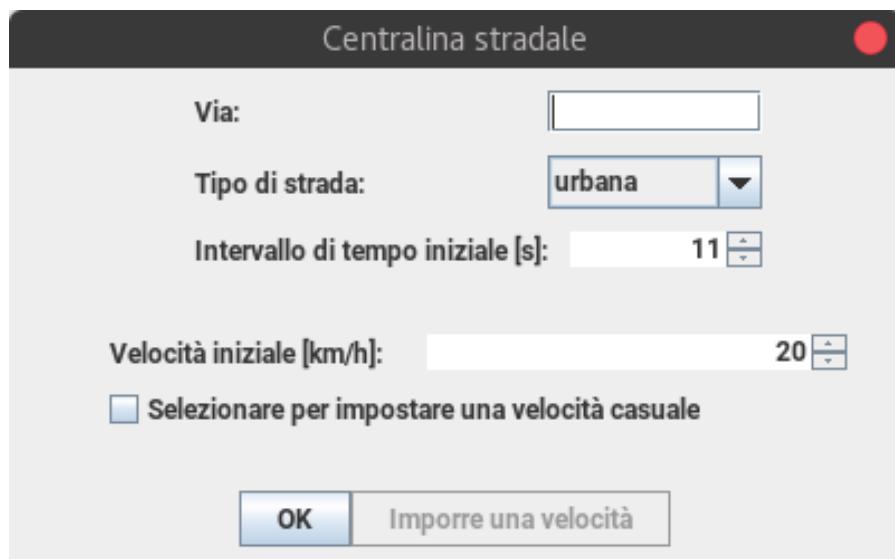


Figura 3: Schermata iniziale della centralina stradale

per esempio da un operatore, nel momento in cui viene installata e che la configurazione rimanga immutata fino ad un riavvio.

L’interfaccia grafica della centralina permette anche di imporre la velocità media registrata in modo da testare il funzionamento dei sistemi anche in assenza di dati reali. In alternativa è possibile utilizzare un simulatore di velocità che genera dei valori casuali che oscillano intorno all’ultimo valore registrato.

Una volta effettuata l’impostazione iniziale, la posizione della centralina (che sarà necessariamente fissa) verrà indicata nella barra del titolo della finestra di impostazione. In ogni caso, sarà possibile osservare un marcitore, di colore variabile in funzione dell’evento di traffico registrato, in corrispondenza della posizione di installazione.

1.2.3 Applicazione mobile

L’applicazione mobile consente all’utente, che deve registrarsi o effettuare il login, di segnalare una coda nella posizione in cui si trova (questa apparirà sotto forma di marcitore sulla mappa, se in esecuzione) e di visualizzare le notifiche relative ad un’area circolare con centro nella posizione rilevata dall’applicazione e raggio di 500 m.

Le notifiche più recenti appariranno nella parte superiore dell’area di notifica e saranno di colori diversi in base alla tipologia, seguendo le stesse convenzioni dei marcatori visualizzati sulla mappa.

1.2.4 Utilizzo di Swing

Le interfacce grafiche sono state realizzate in modo simile. In particolare, si definisce `frame` il contenitore di tutti gli elementi visualizzati



Figura 4: L'applicazione mobile con alcune notifiche

nell'interfaccia grafica.

All'interno di ogni frame possono essere posizionati più pannelli (`JPanel`), che vengono utilizzati come dei contenitori di altri elementi. Ad esempio, la GUI della centralina stradale ha tre pannelli, che contengono i dati di impostazione della centralina, l'impostazione della velocità rilevata (per controllare il funzionamento del sistema complessivo) e i button per confermare le scelte effettuate sulla schermata.

Gli elementi che possono essere utilizzati sono vari. Nel caso di questo progetto sono risultati molto utili gli `spinner` (per selezionare numeri in un dato intervallo), le `label`, i `textfield` (per brevi testi), `textpane` (per testi più lunghi), le `checkbox` (per scegliere tra più opzioni, sotto forma di stringa), i `button` e gli `optionpane` (per visualizzare messaggi di errore e informazioni).

1.2.5 Login e registrazione

Nonostante le interfacce grafiche utilizzate per il login e la registrazione di utenti e amministratori siano uguali, i dati a cui si fa riferimento sono diversi a seconda della tipologia di accesso.

1.3 FORMATO DELLE NOTIFICHE

Le notifiche sono fondamentali per garantire la comunicazione tra i diversi programmi. Per questa ragione le notifiche devono avere un formato standard che sia comprensibile da tutti i programmi.

In particolare, ogni notifica è caratterizzata da un tipo, che è una

stringa. Il formato di questo attributo è il seguente: M oppure S⁴ + velocità + carattere '' + Tipo di evento.

Nel caso della segnalazione della coda da parte delle applicazioni mobili, la velocità non viene rilevata, quindi può essere inserita una qualsiasi stringa prima del carattere '' e questa verrà ignorata.

Un esempio di notifica può essere S10 Traffico elevato e significa che una centralina stradale invia una notifica di traffico elevato registrando una velocità di 10 km/h.

Le notifiche verranno visualizzate sull'applicazione mobile se sono riferite a segnalazioni provenienti da altre applicazioni o centraline stradali nel raggio di 500 m.

Il formato di queste ultime notifiche, però, è più comprensibile per un utente umano. La notifica S10 Traffico elevato proveniente da via Valleggio il giorno 17/12/2018 alle ore 11:11, per esempio, verrà visualizzata come 17/12/2018, 11:11 | Traffico elevato in via Valleggio ad una velocità media di 10 km/h e sarà di colore nero (vedere anche figura 4).

Le notifiche dell'applicazione mobile sono realizzate con un codice HTML, grazie alla libreria `HTMLDocument[VVa]`.

1.4 CASI DI TEST JUNIT

1.5 SINTESI

⁴ M nel caso in cui il dato si origini dall'applicazione mobile, S nel caso in cui provenga da una centralina stradale

BIBLIOGRAFIA

- [VVa] AA. VV. *Class HTMLDocument - javax.swing.text.html.HTMLDocument*. URL: <https://docs.oracle.com/javase/10/docs/api/javax/swing/text/html/HTMLDocument.html>.
- [VVb] AA. VV. *JMapView*. URL: <https://wiki.openstreetmap.org/wiki/JMapView>.
- [VVc] AA. VV. *OpenStreetMap*. URL: <https://www.openstreetmap.org/>.
- [VVd] AA. VV. *Package java.rmi*. URL: <https://docs.oracle.com/javase/10/docs/api/java/rmi/package-summary.html>.
- [VVe] AA. VV. *Package javax.swing*. URL: <https://docs.oracle.com/javase/10/docs/api/javax/swing/package-summary.html>.