

Sistema Multi-Agente per la Gestione delle Emergenze Urbane

Alessio Erasmo, Christian Felicione

7 febbraio 2026

1 Introduzione

Il *CityEmergencyManagementSystem* è un sistema multi-agente sviluppato in DALI secondo la metodologia GAIA per la gestione coordinata di emergenze urbane quali incendi e incidenti. Il sistema coordina automaticamente unità di soccorso attraverso quattro ruoli principali: Dispatcher, DroneScout, AmbulanceUnit e FireRescueUnit.

2 Architettura

Il Dispatcher coordina centralmente il sistema ricevendo segnalazioni e orchestrando le unità operative. I DroneScout sorvegliano il territorio identificando emergenze autonomamente. Le AmbulanceUnit gestiscono il soccorso sanitario mentre le FireRescueUnit intervengono sugli incendi. Gli obiettivi sono salvare vite umane, contenere situazioni di pericolo e garantire risposte coordinate prevenendo escalation.

3 Dinamica Operativa

Le emergenze raggiungono il Dispatcher tramite chiamate esterne o rilevazioni automatiche dei droni. Quando un drone avvista un'emergenza, ne comunica immediatamente localizzazione e tipologia. Se mancano informazioni, il Dispatcher richiede una ricognizione specifica.

Una volta classificata l'emergenza, il Dispatcher assegna le risorse appropriate: AmbulanceUnit per emergenze sanitarie, FireRescueUnit per incendi. Le unità possono accettare confermando la presa in carico o rifiutare specificando il motivo. Al termine dell'intervento comunicano la conclusione delle operazioni, liberando le risorse.

4 Caratterizzazione degli Agenti

| Agente | Comportamento |
|----------------|---|
| Dispatcher | Reattivo: orchestra operazioni senza iniziativa autonoma. |
| DroneScout | Reattivo a richieste; proattivo nel pattugliamento e gestione batteria. |
| AmbulanceUnit | Reattivo a dispatch; proattivo nel richiedere supporto antincendio. |
| FireRescueUnit | Reattivo a dispatch; proattivo nel richiedere supporto sanitario. |

Tabella 1: Comportamenti degli agenti

5 Eventi e Azioni

Il sistema opera secondo un'architettura event-driven. Il Dispatcher gestisce `call_emergency` e `report_emergency`. I droni monitorano `spot_fire` e `spot_accident` oltre al livello di batteria. Le unità operative rispondono a eventi di dispatch gestendo il proprio stato tramite eventi interni come `accept_dispatch`, `refuse_dispatch` e `do_rescue`.

Le azioni specifiche includono `rescue_people` per le ambulanze e `turn_off_fire` per le unità antincendio. Entrambe notificano il completamento tramite `report(emergency_retired)`.

6 Sequence Diagram

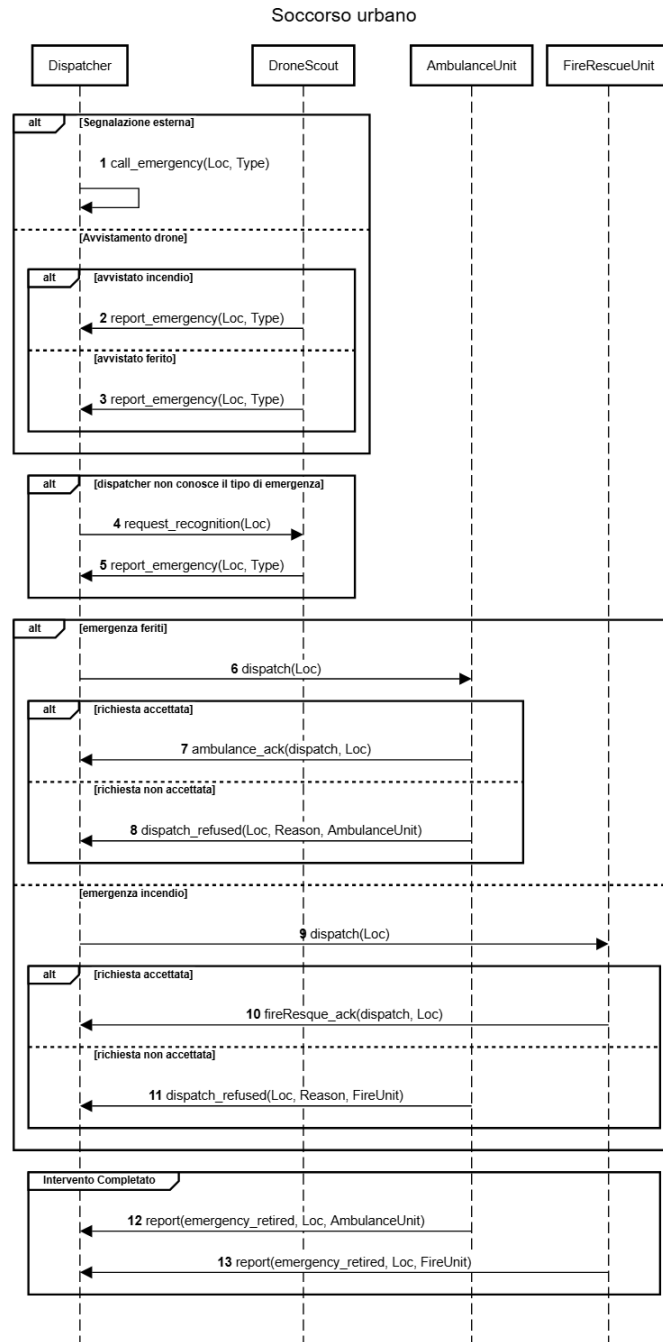


Figura 1: Sequence diagram per la gestione di un'emergenza

7 Class Diagram

Di seguito è riportato il class diagram del sistema multi-agente, che illustra le classi principali, i loro attributi e metodi, nonché le relazioni tra di esse. Il diagramma evidenzia la struttura modulare del sistema, con una chiara separazione dei ruoli e delle responsabilità di ciascun agente.

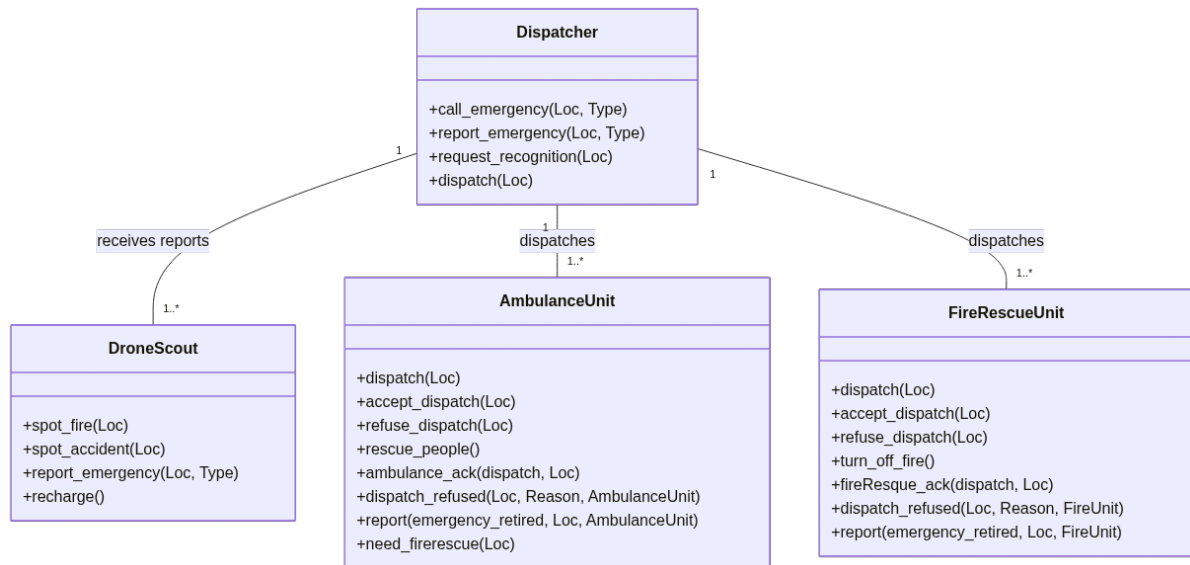
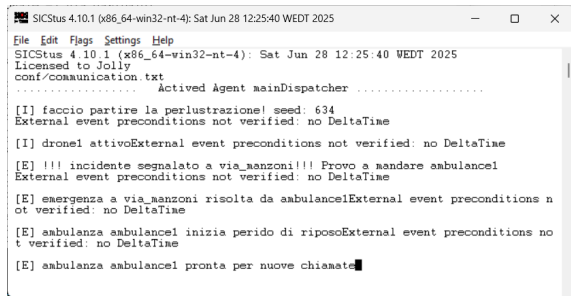


Figura 2: Class diagram del sistema multi-agente

Ogni classe, eccetto il dispatcher, dispone di più istanze, rappresentando le diverse unità operative e droni presenti nella città. Il dispatcher funge da coordinatore centrale, gestendo le comunicazioni e le assegnazioni di compiti tra le varie unità.

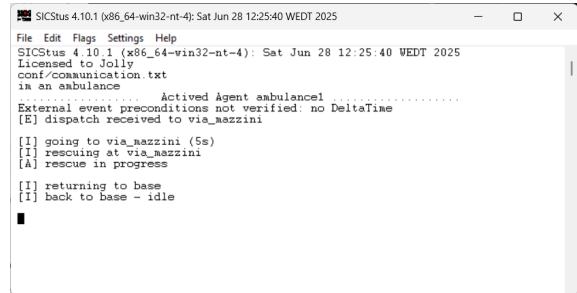
8 Esempio di esecuzione

Dopo l'avvio del sistema, il Dispatcher si attiva e invia un messaggio di **activate** al DroneScout. Il drone inizia a pattugliare l'area, monitorando costantemente per emergenze. Se il drone rileva un'emergenza, invia immediatamente un messaggio al Dispatcher con la posizione. Il Dispatcher quindi classifica l'emergenza e invia un messaggio di **dispatch** all'opportuna unità, che se libera accetta l'incarico e si dirige verso il luogo selezionato per intervenire.



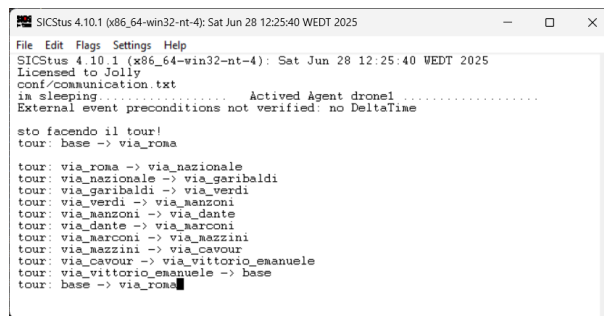
```
SICStus 4.10.1 (x86_64-win32-nt-4): Sat Jun 28 12:25:40 WEDT 2025
File Edit Flags Settings Help
SICStus 4.10.1 (x86_64-win32-nt-4): Sat Jun 28 12:25:40 WEDT 2025
Licensed to Jolly
conf/communication.txt
..... Active Agent mainDispatcher .....
[I] faccio partire la perlustrazione! seed: 634
External event preconditions not verified: no DeltaTime
[I] drone1 attivoExternal event preconditions not verified: no DeltaTime
[E] !!! incidente segnalato a via_manzoni!!! Provo a mandare ambulancel
External event preconditions not verified: no DeltaTime
[E] emergenza a via_manzoni risolta da ambulancelExternal event preconditions not verified: no DeltaTime
[E] ambulanza ambulancel inizia periodo di riposoExternal event preconditions not verified: no DeltaTime
[E] ambulanza ambulancel pronta per nuove chiamate
```

Figura 3: Ciclo di controllo del dispatcher



```
SICStus 4.10.1 (x86_64-win32-nt-4): Sat Jun 28 12:25:40 WEDT 2025
File Edit Flags Settings Help
SICStus 4.10.1 (x86_64-win32-nt-4): Sat Jun 28 12:25:40 WEDT 2025
Licensed to Jolly
conf/communication.txt
in an ambulance
..... Active Agent ambulancel .....
External event preconditions not verified: no DeltaTime
[E] dispatch received to via_mazzini
[I] going to via_mazzini (5s)
[I] rescuing at via_mazzini
[A] rescue in progress
[I] returning to base
[I] back to base - idle
```

Figura 4: Gestione dell'emergenza da parte dell'ambulanza



```
SICStus 4.10.1 (x86_64-win32-nt-4): Sat Jun 28 12:25:40 WEDT 2025
File Edit Flags Settings Help
SICStus 4.10.1 (x86_64-win32-nt-4): Sat Jun 28 12:25:40 WEDT 2025
Licensed to Jolly
conf/communication.txt
is sleeping ..... Active Agent drone1 .....
External event preconditions not verified: no DeltaTime
sto facendo il tour!
tour: base -> via_roma
tour: via_roma -> via_nazionale
tour: via_nazionale -> via_garibaldi
tour: via_garibaldi -> via_verdi
tour: via_verdi -> via_manzoni
tour: via_manzoni -> via_dante
tour: via_dante -> via_marconi
tour: via_marconi -> via_mazzini
tour: via_mazzini -> via_cavour
tour: via_cavour -> via_vittorio_emanuele
tour: via_vittorio_emanuele -> base
tour: base -> via_roma
```

Figura 5: Drone in pattuglia

9 Conclusioni

L'architettura modulare garantisce scalabilità e manutenibilità. La combinazione di comportamenti reattivi e proattivi consente risposte flessibili a scenari complessi, mentre il coordinamento centralizzato assicura coerenza globale. L'implementazione in DALI sfrutta le capacità del linguaggio per eventi, comunicazione inter-agente e ragionamento ibrido, traducendo direttamente i costrutti della metodologia GAIA.