

# Modello di Funzionamento Runtime della Rete

## Introduzione

Il **funzionamento runtime** rappresenta la fase operativa stabile del sistema, in cui la rete di Nodi agisce in modo **distribuito, autonomo ed event-driven**, applicando esclusivamente le regole definite durante la fase di programmazione.

In questa fase non avvengono modifiche di configurazione: il sistema reagisce agli eventi, modifica il mondo fisico e rende osservabile lo stato risultante.

---

## Principio operativo fondamentale

Il modello runtime si basa sulla seguente catena causale:

**Evento → Reazione locale → Stato pubblicato**

La rete non trasporta comandi di attuazione diretti, ma eventi e stati. Le azioni rimangono un dettaglio interno ai Nodi.

---

## Generazione e propagazione degli eventi

Durante il runtime:

- gli Event Generators interni ai Nodi rilevano condizioni locali
- quando una condizione si verifica, viene generato un evento
- l'evento viene pubblicato sulla rete in forma broadcast

L'evento: - non ha destinatari - non implica alcuna azione - rappresenta esclusivamente un accadimento

---

## Reazione agli eventi

Ogni Nodo:

- osserva costantemente gli eventi presenti sulla rete
- confronta ciascun evento con la propria tabella di reazioni
- se una regola corrisponde, esegue una o più azioni locali

La reazione: - è deterministica - è locale al Nodo - non coinvolge coordinamento con altri Nodi

---

## Esecuzione delle azioni

Le azioni:

- sono invocate dal Nodo sui propri Device / Actuators
- producono una modifica del mondo fisico
- possono avere esito positivo o negativo

L'esito dell'azione non viene comunicato come "azione eseguita", ma come **nuovo stato del Device**.

---

## Pubblicazione dello stato

Dopo ogni modifica significativa, il Nodo:

- aggiorna lo stato interno del Device
- pubblica lo stato sulla rete

Lo stato rappresenta: - la condizione reale del dispositivo - eventuali errori o indisponibilità - parametri associati (livello, posizione, timeout)

Lo stato è la **fonte di verità** per l'intero sistema.

---

## Comunicazione periodica dello stato (State Heartbeat)

Per garantire osservabilità e sincronismo:

- ogni Nodo ripubblica periodicamente lo stato dei propri Device
- la frequenza è configurabile e dipende dal tipo di Device

Questa comunicazione periodica consente:

- ricostruzione dello stato globale in qualsiasi momento
  - recupero da perdite di messaggi
  - allineamento di gateway e sistemi esterni
- 

## Osservabilità dal gateway

Un gateway esterno:

- non partecipa alla logica runtime
- osserva eventi e stati
- costruisce una rappresentazione coerente dello stato della casa

Il gateway non deduce lo stato dalle azioni, ma si basa esclusivamente sui messaggi di stato pubblicati dai Nodi.

---

## Interazione con sistemi esterni (MQTT)

Il modello runtime distingue chiaramente tra **eventi generici** e **comandi diretti verso i Device**.

- Gli **eventi generici** (es. pressione di un pulsante, attivazione di una scena) non implicano alcuna azione predefinita e vengono utilizzati per attivare le relazioni configurate in fase di programmazione.
- I **comandi diretti** rappresentano invece una richiesta esplicita di modifica dello stato di un Device specifico.

Un Nodo, anche in assenza di qualsiasi configurazione evento → azione, è sempre in grado di rispondere ai **comandi diretti dei propri Device**.

---

### Comandi diretti ai Device

Ogni Device è, per progetto, **implicitamente iscritto ai propri comandi diretti**.

Ciò significa che:

- un comando come "accendi luce" o "alza tapparella" viene interpretato direttamente dal Nodo proprietario del Device
- non è necessario che tale relazione sia definita in fase di programmazione
- il comportamento di base del Device è sempre disponibile

I comandi diretti consentono il controllo puntuale e immediato dei Device, indipendentemente dalla logica event-driven configurata.

---

### Relazioni evento → azione configurate

La fase di programmazione non definisce il funzionamento di base dei Device, ma **predispone le reazioni agli eventi generici**.

In questa fase vengono configurate relazioni del tipo:

- evento generico → insieme di azioni locali

Ad esempio:

- evento: pressione di un pulsante
- azioni: accensione di una o più luci, movimento di una tapparella

L'evento non codifica cosa deve accadere, ma dichiara esclusivamente che qualcosa è successo. Le azioni vengono applicate solo perché una relazione è stata precedentemente programmata.

---

## Integrazione con MQTT

Nel contesto MQTT:

- i **comandi diretti** provenienti dall'esterno vengono tradotti in richieste di azione verso Device specifici
- gli **eventi esterni** vengono tradotti in eventi generici di rete

Entrambi i meccanismi convivono:

- il controllo diretto garantisce operabilità immediata
  - il modello event-driven garantisce flessibilità e automazione
- 

## Gestione dei comandi esterni

Quando un evento originato da un sistema esterno appare sulla rete:

- i Nodi lo trattano come un evento ordinario
- eventuali regole associate vengono applicate
- le azioni risultanti vengono eseguite localmente

Il sistema non distingue tra eventi interni ed esterni.

---

## Persistenza e ripristino

Ogni Nodo:

- mantiene localmente la configurazione runtime
- conserva lo stato minimo necessario al ripristino

In caso di riavvio:

- il Nodo ripristina il proprio stato
  - ripubblica gli stati dei Device
  - rientra nel funzionamento event-driven
- 

## Proprietà del modello runtime

Il modello runtime garantisce:

- assenza di dipendenze da un controller centrale
  - coerenza osservabile dello stato globale
  - tolleranza a fault e disconnessioni
  - integrazione trasparente con sistemi esterni
-

## Sintesi

Nel funzionamento runtime, la rete opera come un sistema distribuito reattivo in cui:

- gli eventi descrivono ciò che accade
- le reazioni avvengono localmente
- gli stati descrivono ciò che è realmente successo

Questa separazione garantisce robustezza, chiarezza e scalabilità nel tempo.