**ANALISI STRUMENTALE**

L’obiettivo di Serra 4.0 è creare un sistema completo per il monitoraggio e la visualizzazione di dati in tempo reale provenienti dai sensori di una serra automatizzata. Utilizzando Node-RED per l’acquisizione e l’elaborazione dati, PostgreSQL come database per l’archiviazione strutturata e Grafana per la visualizzazione e l’analisi interattiva, si ottiene una soluzione scalabile, modulare e personalizzabile per ottimizzare la gestione ambientale della serra.

**PostgreSQL:** è un database relazionale open-source, scelto per la sua affidabilità, performance e supporto a query complesse.

* Configurazione: Database strutturato con tabelle per dati sensoriali rilevati in Serra 4.0 (es. temperatura, umidità, luminosità, timestamp).
* Accesso: Node-RED interagisce con PostgreSQL tramite il nodo node-postgres.
* Ruolo: Conservazione storica dei dati raccolti, supporto a query aggregative per Grafana.

**Node-RED:** è una piattaforma di programmazione visuale per connettere hardware, API e servizi online.

* Flusso principale:
  + Acquisizione dati da sensori MQTT installati in Serra 4.0.
  + Trasformazione e normalizzazione dei dati (es. conversione unità).
  + Inserimento dei dati nel database PostgreSQL tramite nodo apposito.
* Vantaggi: Facilità di creazione di flussi personalizzati senza programmazione complessa.

**Grafana:** è uno strumento open-source per la creazione di dashboard interattive.

* Configurazione: Collegamento diretto a PostgreSQL come fonte dati per Serra 4.0.
* Dashboard: Creazione di grafici a linee e gauge per visualizzazione in tempo reale l’andamento di temperatura, umidità e altri parametri ambientali.
* Funzionalità aggiuntive: Configurazione alert per notifiche su valori fuori soglia.

**Mosquitto**: è un broker MQTT open-source leggero, utilizzato per la trasmissione dei dati tra i sensori della serra e la piattaforma di elaborazione.

* **Ruolo**: Funziona da intermediario tra i sensori IoT e Node-RED, ricevendo i messaggi pubblicati dai dispositivi e rendendoli disponibili ai sottoscrittori.
* **Vantaggi**:
  + Basso consumo di risorse, ideale per dispositivi embedded e ambienti a bassa latenza.
  + Supporto al protocollo MQTT, standard de facto per la comunicazione M2M in scenari IoT.
  + Possibilità di configurare autenticazione e crittografia per la sicurezza dei dati.
* **Integrazione**: I sensori di Serra 4.0 pubblicano i dati su specifici topic MQTT gestiti da Mosquitto. Node-RED si sottoscrive a questi topic per acquisire i dati in tempo reale.

**WebSocket**: è un protocollo di comunicazione full-duplex che consente uno scambio bidirezionale continuo di dati tra client e server.

* **Ruolo**: Permette la trasmissione in tempo reale dei dati acquisiti da Node-RED verso interfacce web o pannelli utente, senza la necessità di refresh o polling continuo.
* **Vantaggi**:
  + Comunicazione efficiente con latenza minima, ideale per il monitoraggio in tempo reale.
  + Riduzione del carico di rete rispetto alle tradizionali richieste HTTP.
  + Supporto nativo in Node-RED tramite nodi dedicati.
* **Integrazione**: Node-RED invia i dati elaborati tramite WebSocket ai client (es. dashboard web), garantendo aggiornamenti immediati su variazioni nei parametri ambientali della serra.

**Mosquitto:**

**Descrizione generale**  
Mosquitto è un broker MQTT open-source progettato per abilitare comunicazioni efficienti tra dispositivi IoT mediante il paradigma *publish/subscribe*. È caratterizzato da leggerezza, affidabilità e semplicità di configurazione, ed è ampiamente usato in sistemi distribuiti e ambienti embedded.

**Ruolo nel sistema Serra 4.0**  
Nel contesto della serra automatizzata, Mosquitto gestisce la trasmissione dei dati tra i sensori IoT e Node-RED. I dispositivi installati nella serra (es. per temperatura, umidità, luminosità) pubblicano dati su specifici topic MQTT. Mosquitto riceve questi messaggi e li rende disponibili ai sottoscrittori, tra cui Node-RED, che li elabora e li inoltra al database.

**Funzionamento e architettura**

* I sensori agiscono come **publisher**, inviando messaggi a topic MQTT.
* Node-RED agisce come **subscriber**, ricevendo i dati in tempo reale.
* Il broker Mosquitto gestisce le connessioni, inoltra i messaggi e può essere configurato per garantire la persistenza e la sicurezza della comunicazione.

**Caratteristiche tecniche**

* **Protocollo supportato**: MQTT v3.1, v3.1.1 e v5.
* **Qualità del servizio (QoS)**: supporta tre livelli (0, 1, 2) per la garanzia di consegna dei messaggi.
* **Configurabilità**: tramite file mosquitto.conf, consente la definizione di listener multipli, autenticazione, crittografia TLS, ACL (Access Control List), log personalizzati e persistenza.
* **Compatibilità**: funziona su Linux, Windows e dispositivi embedded.

**Vantaggi**

* **Efficienza**: basso consumo di risorse, adatto a dispositivi con capacità limitate.
* **Affidabilità**: gestione robusta delle connessioni e dei messaggi.
* **Sicurezza**: supporto a crittografia TLS/SSL, autenticazione utente, autorizzazioni granulari.
* **Scalabilità**: può essere distribuito in architetture IoT complesse, anche con migliaia di dispositivi.
* **Integrazione diretta**: interoperabilità ottimale con Node-RED, facilitando la costruzione di pipeline dati in tempo reale.

**Esempio d’uso in Serra 4.0**

* I sensori pubblicano su topic come serra/temperatura, serra/umidita.
* Mosquitto inoltra questi messaggi a Node-RED, che li elabora e li memorizza in PostgreSQL.
* Grazie al supporto WebSocket, i dati possono essere inviati in tempo reale a dashboard web tramite Node-RED.

**WebSocket**

**Descrizione generale  
WebSocket è un protocollo di comunicazione bidirezionale full-duplex, basato su TCP, che consente uno scambio continuo di dati tra client e server attraverso una connessione persistente. È particolarmente utile per applicazioni che richiedono aggiornamenti in tempo reale, come il monitoraggio ambientale della serra.**

**Ruolo nel sistema Serra 4.0  
In Serra 4.0, WebSocket viene utilizzato per trasmettere i dati elaborati da Node-RED verso interfacce utente come dashboard web o pannelli di controllo. Questo consente la visualizzazione immediata dei dati, evitando ritardi o la necessità di ricaricare la pagina (polling HTTP).**

**Funzionamento e architettura**

* **Una volta elaborati i dati da Node-RED (ricevuti via MQTT), questi vengono inviati tramite nodi WebSocket a client web connessi.**
* **La connessione WebSocket rimane aperta finché client e server lo desiderano, permettendo scambi continui di informazioni in entrambe le direzioni.**

**Caratteristiche tecniche**

* **Protocollo: WebSocket (RFC 6455), basato su TCP.**
* **Connessione persistente: a differenza di HTTP, la connessione rimane attiva dopo il primo handshake.**
* **Integrazione: Node-RED dispone di nodi websocket in e websocket out per la gestione semplificata dei flussi dati.**
* **Supporto client: compatibile con tutti i browser moderni e facilmente integrabile in interfacce JavaScript.**

**Vantaggi**

* **Aggiornamento in tempo reale: ideale per sistemi di monitoraggio.**
* **Efficienza di rete: riduce il carico rispetto a soluzioni basate su richieste periodiche.**
* **Bidirezionalità: consente anche al server (Node-RED) di inviare aggiornamenti proattivamente.**
* **Integrazione web: nativamente supportato da browser, perfetto per dashboard e interfacce personalizzate.**

**Esempio d’uso in Serra 4.0**

* **Node-RED riceve i dati ambientali via MQTT e li normalizza.**
* **I dati vengono trasmessi tramite WebSocket a una dashboard web.**
* **L’utente può visualizzare in tempo reale le variazioni di temperatura, umidità, luminosità e ricevere alert visivi o sonori.**