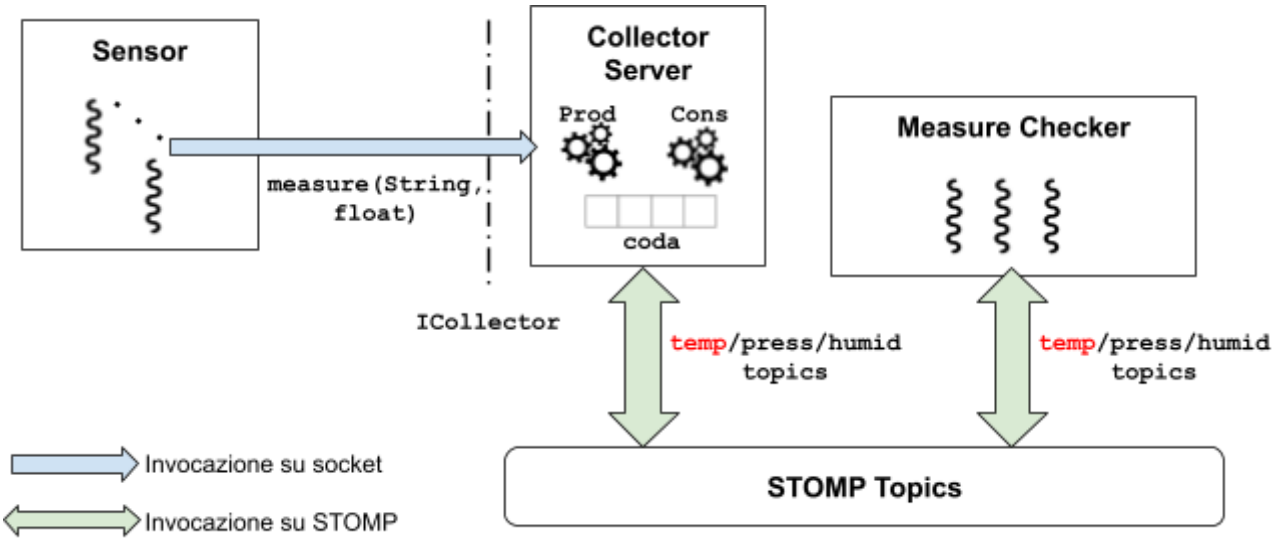
Università degli Studi di Napoli Federico II
Esame di Advanced Computer Programming

Proff. De Simone, Della Corte

Prova pratica del giorno 25/07/2025
Durata della prova: 120 minuti

Testo della prova



Il candidato implementi un sistema distribuito in **Python** per la gestione di misurazioni di sensori IoT basato su **Socket** e **STOMP**. Il sistema è caratterizzato dai seguenti componenti.

Sensor. E' un client multithread che genera le misurazioni da inviare al **Collector Server**. L'invio di una misurazione consiste nell'invocazione del metodo *void measure(String, float)* specificato nell'interfaccia **ICollector**. La richiesta è caratterizzata da 1) **tipo** (*String*), ossia la tipologia di sensore (temperatura, pressione, umidità); 2) **value** (*float*), ossia il valore misurato dal sensore e da inviare. Il client Sensor crea 5 thread sensori, che invocano il metodo *measure* per 5 volte (attendendo 1 secondo tra le invocazioni). Per ciascun sensore, *tipo* è generato in maniera casuale scegliendo una tra le tipologie di sensore (temperatura, pressione, umidità), come anche il valore del campo *value*.

Collector Server. Fornisce l'interfaccia *ICollector* e il relativo metodo *void measure(String, float)*. All'invocazione del metodo *measure*, si avvia un processo produttore, il quale inserisce in una **coda di dimensione *N = 6*** (*process-safe* e che implementi il problema del produttore/consumatore) una stringa che concatena sia la stringa del parametro *tipo* che il valore del parametro *value* (ad es., `temperatura-19.7`). I dati inseriti nella coda sono consumati da un processo consumatore avviato al lancio del ***Collector Server***. Quando sono disponibili N elementi nella coda, il processo consumatore deve svuotarla e inviare tutti i dati consumati al *Measure Checker* attraverso *N messaggi STOMP*. I messaggi vengono inviati su tre topic STOMP distinti (*temp, press, humid*) **a seconda della tipologia di sensore. Notare che l'invio dei messaggi sul topic *temp* deve essere transazionale (gestire opportunamente in caso di abort).** Per i topic *press* e *humid* non c'è bisogno della transazionalità.

Measure Checker. E' un modulo che crea 3 thread, uno per ogni tipologia di misurazione. Ogni thread crea una subscription durabile sul topic STOMP opportuno (e.g., thread check temperatura farà la subscription al topic *temp*). Alla ricezione di ciascun messaggio, il listener STOMP di tali messaggi estrae il contenuto del messaggio, memorizza opportunamente i dati, ed effettua il calcolo di media, minimo, e massimo da stampare a video.

Il candidato utilizzi proxy-skeleton con socket UDP per la comunicazione tra Sensor e Collector Server, e Topic STOMP per quella tra Collector Server e Measure Checker. A tal fine, il candidato predisponga le opportune interfacce e le classi Proxy-Skeleton. Si utilizzi inoltre skeleton per delega per il Collector Server.