|  |  |
| --- | --- |
| Discover the Top 10 IoT Applications in 2020 | Transmedia ... | Progetto Finale software Engineering for autonomous system |
| Nome progetto: | Termocoperta intelligente per terapia intensiva |
|  |  |

Questo progetto riguarderà lo sviluppo di una termocoperta intelligente in grado di adattare la temperatura della coperta in base alle necessità del paziente.

|  |  |  |
| --- | --- | --- |
| Partecipanti | | |
| Nome | Cognome | Matricola |
| Alessandro | Sallese | 264081 |

|  |
| --- |
| Idea |

L’idea consiste nello sviluppo di una termocoperta in grado di adattare la temperatura in base all’obiettivo richiesto dal personale sanitario ed inoltre integrandolo ad un sistema di ML in grado di adattarsi alle richieste passate.

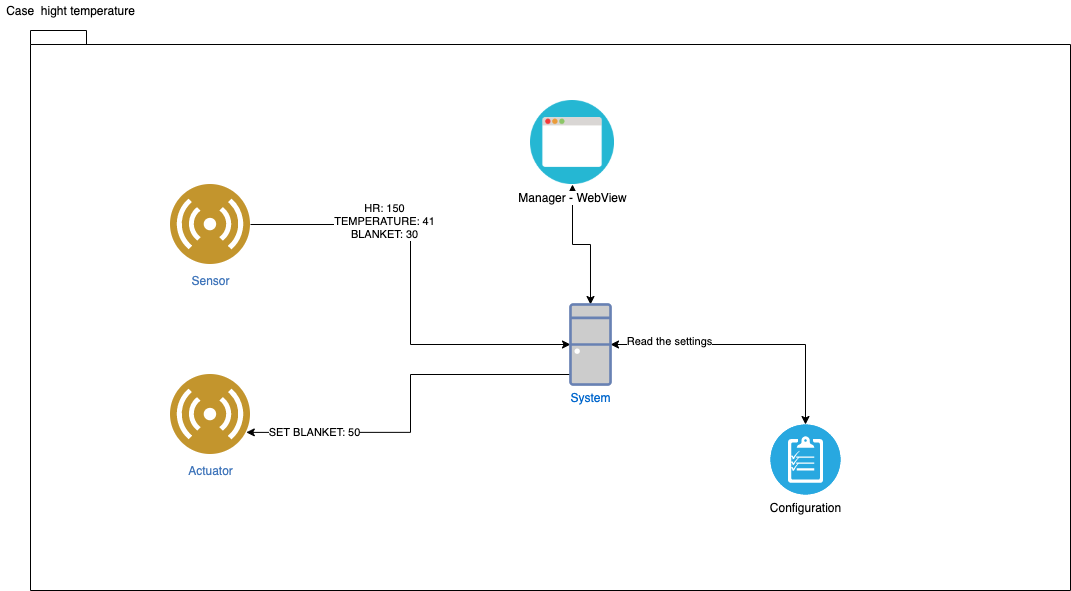
Lo scopo è quello di ridurre gli interventi del personale sanitario nella regolazione della temperatura del paziente, attraverso l’adozione di comportamenti corretti e standardizzati con aumento o diminuzione della temperatura corporea.

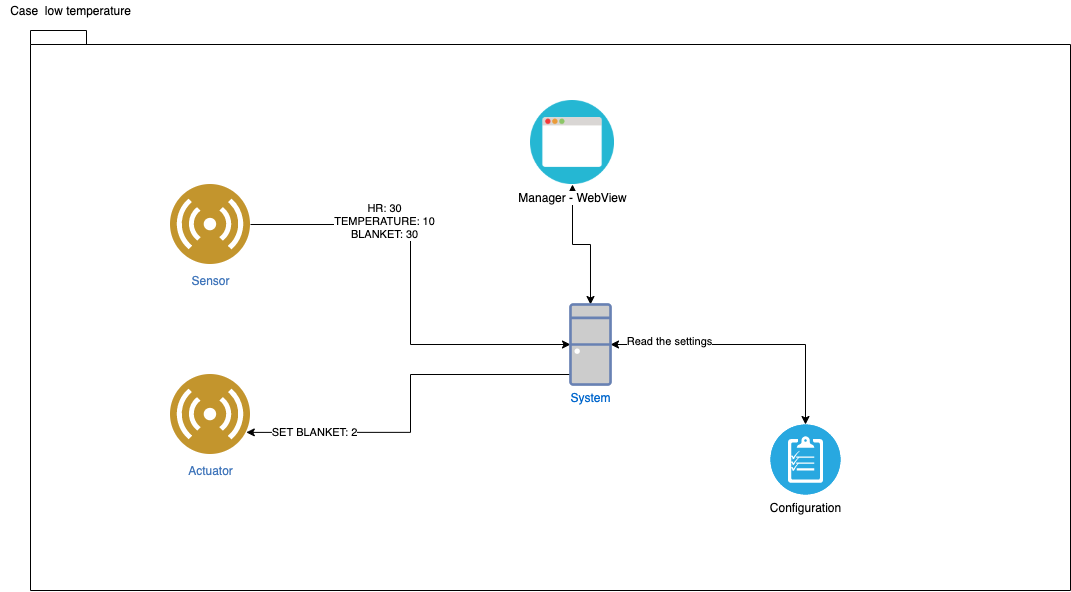
Le casistiche analizzate sono quelle di febbre molto alta oppure di ipotermia o di semplice mantenimento della temperatura corporea.

Per poter capire in quale situazione ci troviamo il sistema analizza ad intervalli regolari i parametri vitali del paziente in modo da poter adottare la strategia migliore.

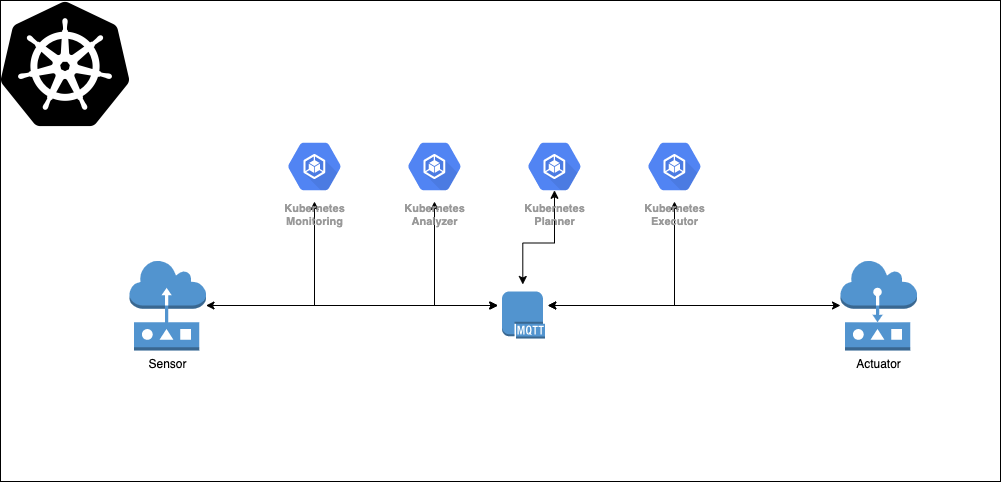
Ad esempio, un paziente sottoposto ad intervento chirurgico può andare incontro a numerose complicanze; tra queste vi è il rischio, se non adeguatamente riscaldato durante l’intervento, di soffrire delle conseguenze dovute alla prolungata e trascurata ipotermia intraoperatoria. Gli effetti avversi causati dall’ipotermia sono numerosi e ben documentati in letteratura e possono avere ripercussioni sia durante l’intervento (aumentate perdite e necessità di trasfusioni) che nel periodo post-operatorio (maggior suscettibilità alle infezioni e maggior tempo di permanenza in terapia intensiva).

Per poter riuscire in questo obiettivo il progetto eseguirà uno schema MAPE-K attraverso l’ausilio delle seguenti tecnologie: Kubernetes, Mqtt, Sklearn, Python, Angular.





|  |
| --- |
| MAPE-K |
| Il progetto si sviluppa secondo il seguente schema Monitoring: Il monitoraggio avverrà attraverso la simulazione di alcuni sensori di temperatura e di frequenza cardiaca. Nel caso reale la temperatura sarà monitorizzata utilizzando una sonda cutanea oppure in caso di patologie infettive di catetere vescicale con sonda di temperatura. I dati verranno comunicati al componente di ANALISI tramite un broker Mqtt che, inoltre, fungerà da canale comune per la comunicazione tra i vari componenti. Analyzing: La fase di analisi è collegata tramite il componente di monitoring da un broker Mqtt utilizzato per la trasmissione dei dati “catturati dai sensori”. Poiché in questa fase il sistema deve valutare la situazione dell’environment che aggiornerà la rete neurale.  La rete neurale sarà implementata attraverso l’utilizzo di Python con l’uso delle librerie Sklearn.  A tal proposito il training della rete neurale avverrà tramite un dataset con le abitudini di “temperatura” del paziente. Il risultato dei dati analizzati verrà trasmesso dall’ANALYZER al PLANNER sempre tramite Mqtt. |
| Planning In base alla fase di analisi, il Planner avrà il compito di elaborare la migliore strategia da attuare sia in base all’analisi realizzata precedentemente e sia attraverso gli standard vitali da rispettare.  La definizione di febbre è ancora oggi arbitraria. La temperatura corporea media in volontari sani, ha presentato un range da 35.6°C a 38.2 °C quando misurata a livello della mucosa orale. La Società di Medicina Critica e la Società di Malattie Infettive Americana in una recente consensus hanno definito febbre una temperatura corporea superiore a 38.3°C. Per quanto riguarda lo stato di ipotermia considereremo il caso di temperature al di sotto dello standard appena definito.  La comunicazione all’esecutore avverrà attraverso un canale di Mqtt e invierà la decisione presa dal Planner. |
| **Executing:**  Si occupa tramite attuatori dell’esecuzioni di azioni volte ad un cambiamento nell’environment (se il ciclo MAP ha prodotto e “ritenuto opportuno” una strategia). I vari comandi da eseguire verranno trasmessi agli attuatori direttamente attraverso un canale apposito. |
| **Kwnoledge:**  Tale operazione avverrà tramite Kubernetes che permetterà di conoscere costantemente la situazione delle componenti e attuare delle politiche di recovery e monitoring delle varie componenti.  Tramite gli Health-Checker si potrà monitorare lo stato di salute delle varie componenti e nel caso di risposta negativa si potrà procedere al riavvio del pod. |



# Nello sviluppo

## Introduzione

Per la realizzazione del progetto si è scelto di adottare Kubernetes affinché ogni servizio fosse indipendente e facilmente monitorabile dall’ecosistema a Pod.

Per ogni servizio deployato Kubernetes permette di monitorare e settare tutte le risorse necessarie. Tale sistema permette inoltre, attraverso strategie definite nei file di demployment, di scalare, aggiornare o far ripartire in modo autonomo i vari servizi all’interno di Kubernetes. L’obiettivo è quello di permettere al sistema di continuare autonomamente dopo una fase di training durante la fase manuale.

## Sensor-Monitoring

Per motivi di esemplificativi la fase di Monitoring è compresa nella realizzazione dei sensori ove si è scelto di sviluppare un servizio di generazione randomica di valori rilevati dai sensori. Tali valori, ad ogni modo, si muovono in un range compatibile ad una casistica reale. Inoltre, in questa fase, vengono raccolti tramite interfaccia web gli eventuali input dell’operatore che interagisce con il sistema. Pur essendo due componenti separate identificate con 2 Pod seprati a livello concettuali, essi fanno parte della fase di Monitoring.

Tutti i valori generati vengono inviati al broker di messaggi Mqtt che a sua volta inoltra ai servizi interessati i valori generati.

Essendo sviluppato all’interno di Kubernetes anch’esso prevede un file di specifica in grado di gestire al meglio la fault tollerance di tale servizio.

## Analyzing

In questa parte di approccio progettuale, il sistema provvede ad analizzare le temperature selezionate durante le fasi precedenti di studio. Infatti, tale parte di sfotware contenuto nel POD Analyzing, parte da una conoscenza pregressa contenuta nel dataset health.

Grazie alla rete neurale realizzata a partire da questo dataset, si ha infatti la possibilità di prevedere il settaggio migliore della termocoperta.

Il risultato di tale analisi viene trasmesso al broker di messaggi Mqtt che si occuperà di pubblicare tale messaggio a tutte le componenti che lo richiedono.

## Planning

A partire, tutte le informazioni raccolte il sistema, prende in carico l’interazione dell’operatore e le rilevazioni dei sensori oltre che al risultato della fase di analisi.

In base anche ai valori settati nel Settings.yaml e in base ai parametri decisi dall’operatore, il sistema pianifica la soluzione da adottare prevedendo i nuovi settaggi da impartire alla parte del sistema di Executing.

Tutti i risultati ottenuti da questa fase vengono pubblicati al broker di messaggi Mqtt.

## Executing

In questa parte del sistema il POD deployato andrà a recuperare tutte le impostazioni decise dal Planner per poi ripartirle ai vari esecutori.

Per motivi esemplificativi attuatori e alcuni sensori si trovano sullo stesso canale.

## Knowledge

In questa ultima parte del sistema, Kubernetes avrà la conoscenza dello stato di tutte le componenti. In caso di malfunzionamenti e/o errori imprevisti sarà in grado di gestirli e adottare la soluzione più opportuna.

In questo caso specifico è stato scelto di attuare solamente il riavvio automatico per motivi esplicativi.

Di seguito viene riportato uno scorcio della strategia di verifica dello stato di salute del POD che viene, come detto in precedenza, controllato e gestito da Kubernetes.

# Strategie adottate

## Health checker

Per verificare se un contenitore in un pod è sano e pronto a servire il traffico, Kubernetes fornisce una serie di meccanismi di controllo della salute. I controlli di salute, o probes come sono chiamati in Kubernetes, sono eseguiti dal kubelet per determinare quando riavviare un contenitore (per livenessProbe) e utilizzati da servizi e distribuzioni per determinare se un pod dovrebbe ricevere traffico (per readinessProbe).

Immagine che contiene testo

Descrizione generata automaticamente

Definendo nel deployment, questa specifica, il sistema si occuperà di effettuare dei controlli sullo stato di salute del pod e di riavviarlo in caso di mancata risposta.

# Sequence Diagram

