Robotic Arms Management System (R.A.M.S.)

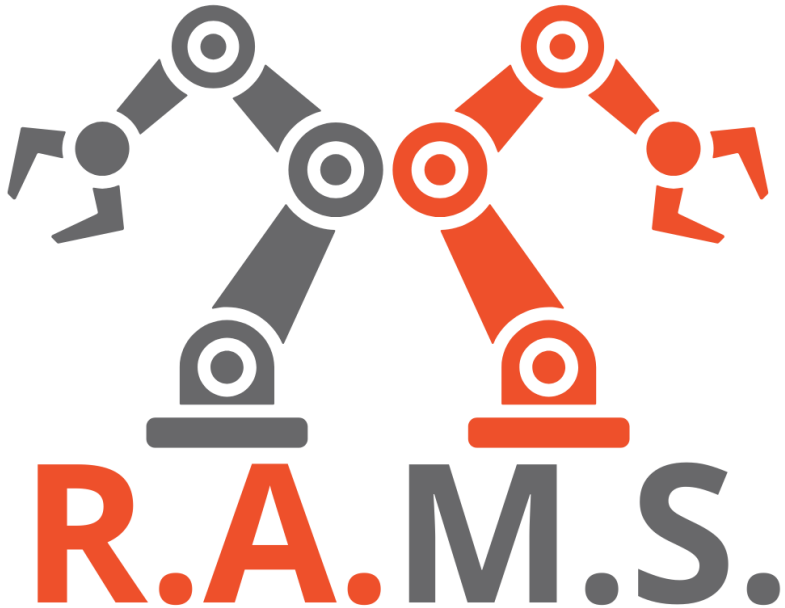
Matteo Tagliavini Vittorio Nutricato

IOT and 3D Intelligent Systems – Roberto Vezzani

Presentazione del progetto

Presentazione del progetto
Panoramica

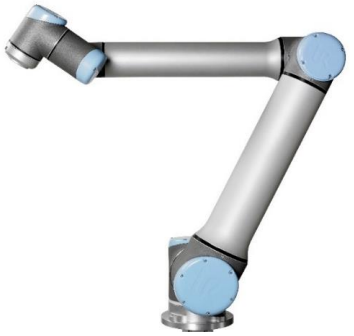
Un sistema di monitoraggio e controllo intelligente che renderà più facile la gestione di una collezione di Robot. Il sistema permette un incremento e miglioramento della produzione, centralizzando il processo decisionale ma allo stesso tempo permettendo ai Robot di lavorare in completa autonomia.



Presentazione del progetto
Cobot

I **Cobot** (collaborative robot), sono Robot collaborativi, progettati per condividere lo spazio di lavoro con esseri umani senza barriere protettive intorno. Non sostituiscono gli operatori ma li affiancano, interagendo in modo funzionale all'esercizio di un compito.

Spesso si configurano come uno o due bracci robotici con almeno sei gradi di libertà, ovvero capaci di compiere movimenti su sei assi:
tre di traslazione e tre di rotazione.



Presentazione del progetto
Funzionamento

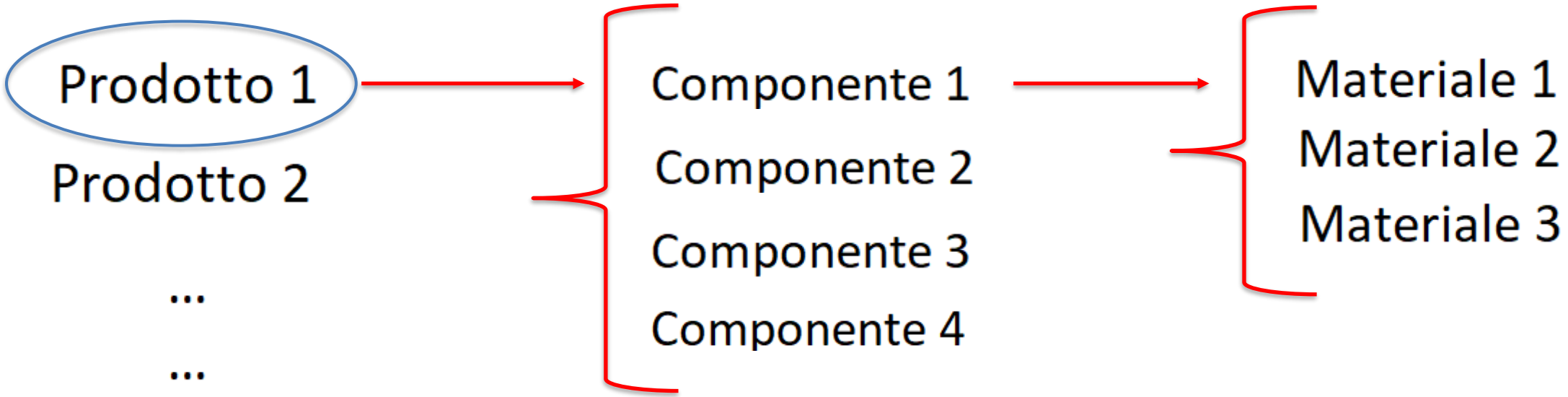
Il sistema **R.A.M.S.** ha una duplice utilità:

- Permette il monitoraggio delle condizioni dell'ambiente di lavoro, in particolare temperatura e pressione, di dove si trova il Cobot.

- Offre la possibilità di poter assegnare da remoto delle lavorazioni ad ogni Cobot, sulla base delle risorse e disponibilità di materiali nel proprio magazzino locale. In questo modo la produzione viene divisa e portata avanti in parallelo, risparmiando tempo sulla completa realizzazione del prodotto.

Presentazione del progetto
Materiali → Componenti → Prodotto

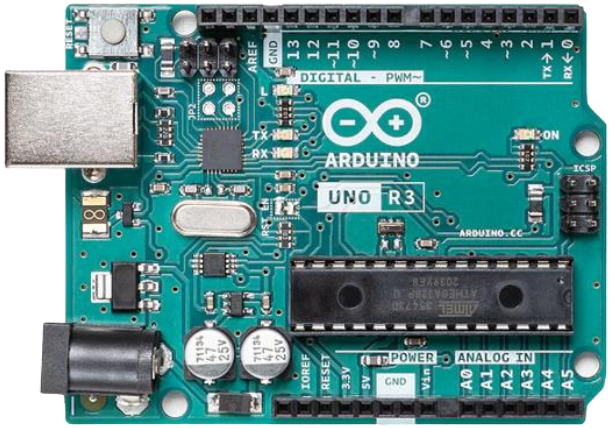
Ogni Cobot della nostra flotta è programmato per poter eseguire diversi task, utilizzando i materiali a disposizione nel proprio magazzino.

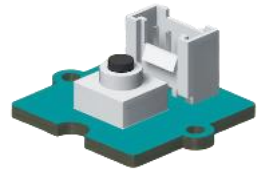


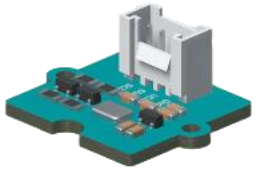
Presentazione del progetto
Componenti del kit

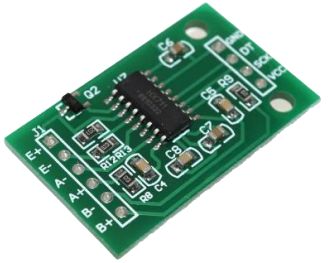
Non si dovrà sostituire o modificare nessun Cobot. Semplicemente sarà necessario installare il kit e configurarlo in modo tale da poter condividere sul server online i dati raccolti.
Il kit è composto da:

- un **microcontrollore**
- un **sensore di temperatura e pressione**
- 3 **sensori di peso**
- un **bottone**









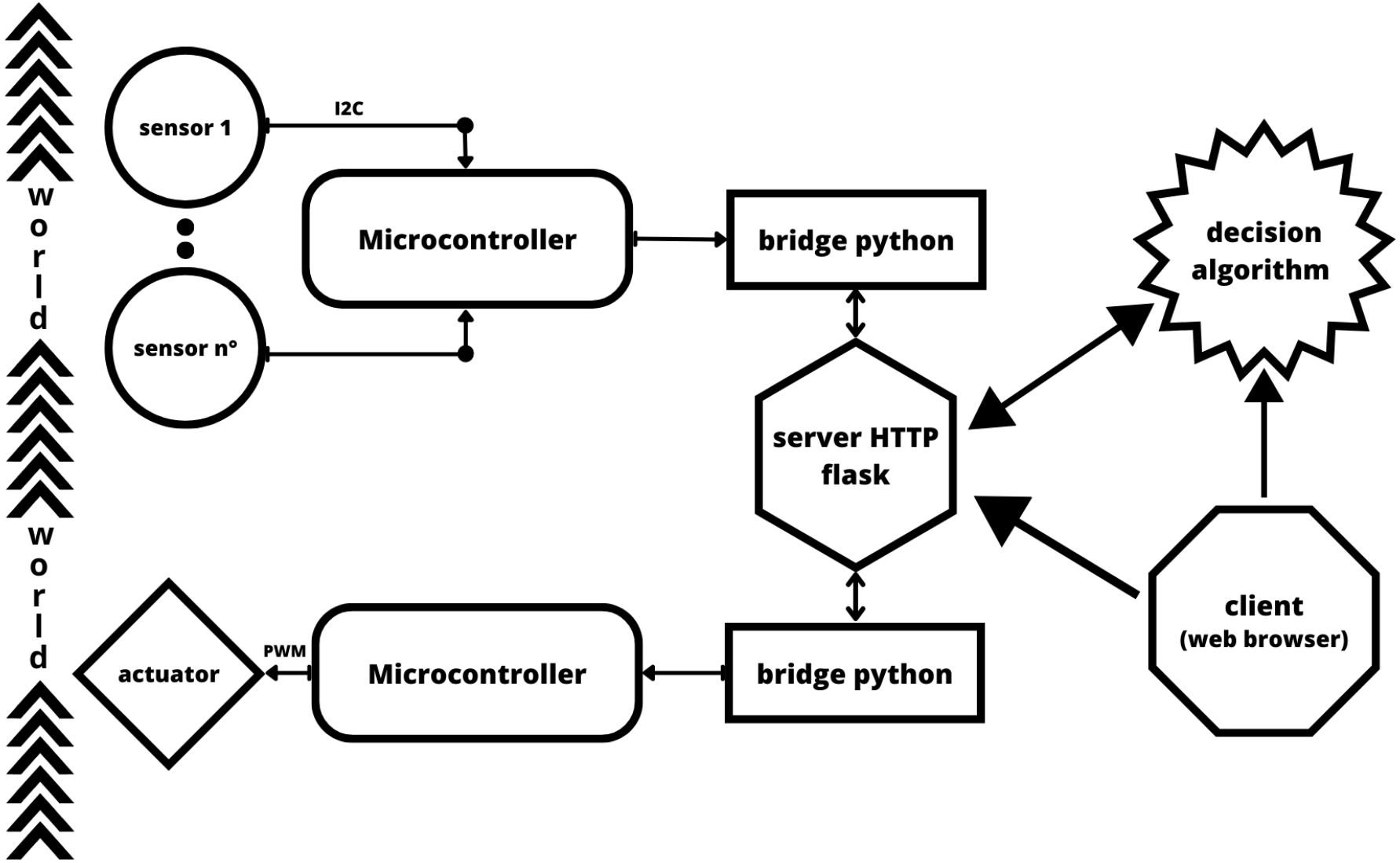
Architettura del sistema

Architettura del sistema

- Il cliente avrà modo di acquistare uno o più kit R.A.M.S. dimensionati per le proprie esigenze, con la sensoristica adeguata alla tipologia di braccio robotico a sua disposizione e tanti sistemi di controllo (bridge e microcontrollori) quante le postazioni di lavoro che vorrà inserire nel Robotic Arms Management System.

- In particolare ogni postazione sarà formata da due coppie bridge-microcontrollore, una che «osserverà» l'ambiente di lavoro per creare il **digital twin**, l'altra che invece riporterà nel mondo reale le decisioni prese dall'algoritmo attraverso l'attuatore, ovvero il braccio robotico.

Architettura del sistema

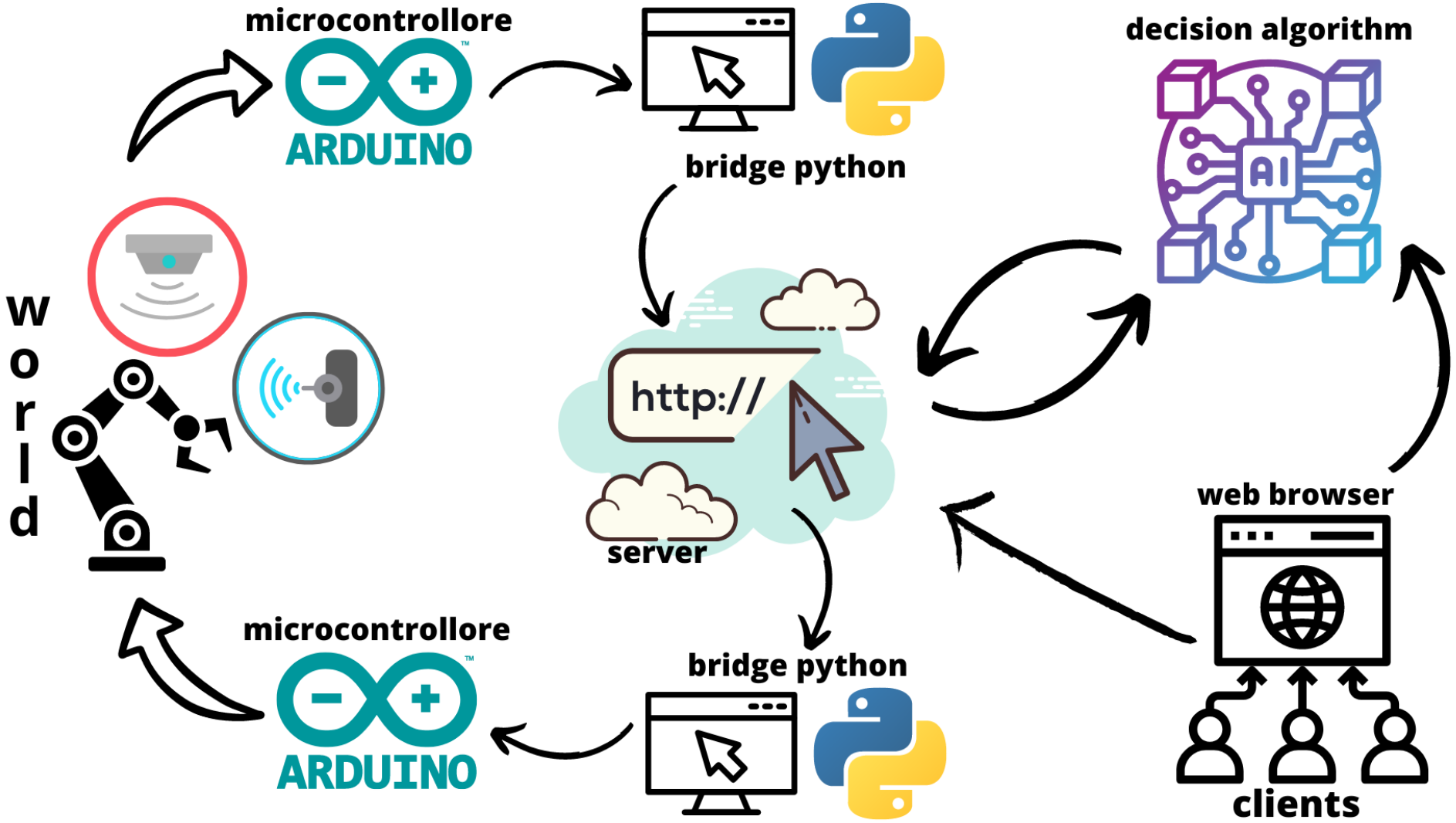


Architettura del sistema

- Sul cloud saranno registrati tutti i dati relativi alle varie postazioni di lavoro inserite nel sistema. Tramite qualunque browser il cliente potrà sfruttare la web application per controllare lo stato dell'intero sistema, disponibilità di materiali e sistemi attivi o disponibili in tempo reale.

- Tramite la web application, oltre che da terminale su richiesta del cliente, sarà anche possibile fare richiesta di lavorazioni al sistema, in particolare in fase di distribuzione del prodotto, quest'area andrà protetta tramite autenticazione del cliente per motivi di sicurezza.

Architettura del sistema



Architettura del sistema

L'architettura prototipale realizzata modella completamente l'architettura di una postazione di lavoro, associata ad un braccio robotico, mostrata dal progetto:

- Come **microcontrollori** sono stati utilizzati un *Arduino Uno* ed un *Arduino Mega 2560* (il secondo scelto per disponibilità). Abbiamo deciso di separare la gestione dei sensori e dell'attuatore perché in fase realizzativa del prodotto, la movimentazione dei bracci robotici potrà essere gestita direttamente tramite protocollo HTTP con comandi mirati alla realizzazione dei singoli prodotti diretti al robot.

Architettura del sistema

- Sono stati inseriti nel prototipo **un sensore barometrico** (BMP280) per la misurazione di temperatura e pressione atmosferica, **un bottone** per determinare la fine della lavorazione sulla postazione, tre celle di carico che monitorano lo stoccaggio dei materiali di lavorazione e due **servo motori** (SG90) che simulano il movimento di un braccio robotico.
- Come **bridge** sono stati usati dei PC con script Python. È stato anche creato un bridge che simula la presenza di nove postazioni di lavoro oltre al prototipo realizzato, anch'esso in comunicazione con il server.
- Il **server HTTP** è stato creato tramite il framework Flask in Python.

Specifiche del microcontrollore e comunicazioni tra i vari componenti

Specifiche del microcontrollore

I due microcontrollori, alimentati direttamente dai PC che fungono anche da bridge, le tre celle di carico, il sensore di temperatura e pressione, il bottone di fine lavorazione e i due servo motori sono posizionati sul prototipo di quella che sarà la postazione di lavoro, ovvero il kit R.A.M.S venduto al cliente che si occuperà di:

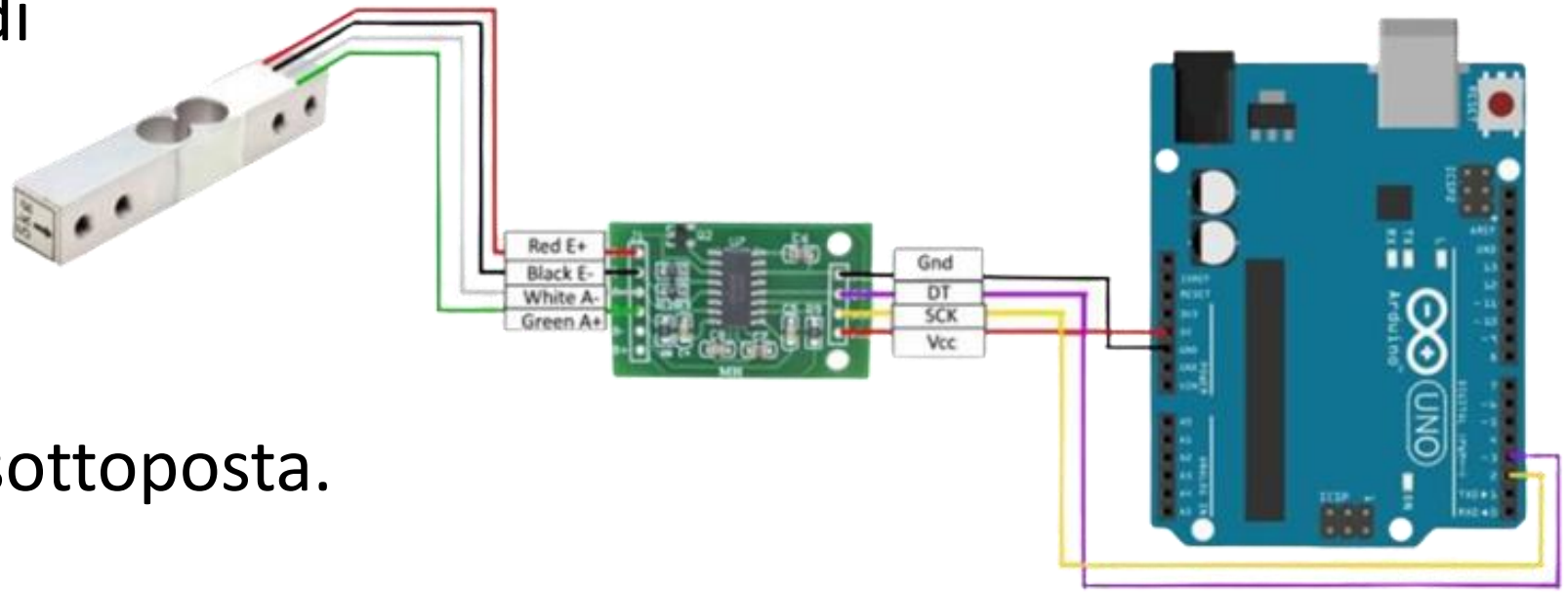
- **Trasmettere** al bridge lo stoccaggio dei materiali di lavorazione tramite il loro peso, il valore della temperatura e pressione dell'ambiente e lo stato di avanzamento della lavorazione tramite la pressione del bottone.
- **Ricevere** dal bridge eventuali indicazioni di lavorazione da eseguire attivando i due servomotori ancorati al sistema di monitoraggio.

Comunicazione Micro-Sensori-Attuatori

- **La comunicazione microcontrollore-bridge** e viceversa è permessa dalla comunicazione seriale (connessione usb) tramite protocollo UART (Universal Asynchronous Receiver Transmitter) che permette di collegare un TX e un RX ad una distanza ridotta.
- **La comunicazione del sensore barometrico** con il microcontrollore avviene tramite protocollo asincrono I2C basato su due linee SDA e SCL.
- **La comunicazione del secondo microcontrollore con i due servomotori** avviene tramite canale PWM di Arduino, un segnale a frequenza costante ma con duty cycle variabile tra 0 e 100%.

Specifiche delle celle di carico

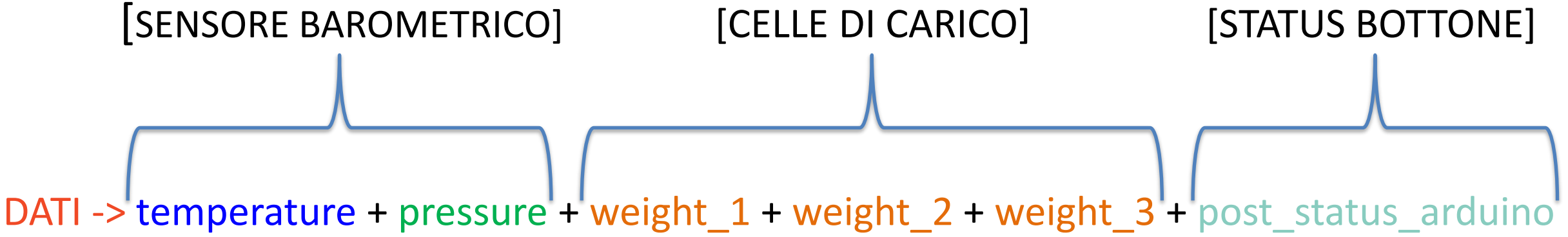
- Le celle di carico sfruttano la variazione di resistenza elettrica che alcuni materiali manifestano quando sottoposti a compressione.
- L'amplificatore per celle di carico utilizza il chip HX711, composto da un amplificatore a guadagno variabile e da un convertitore analogico-digitale. La variazione di resistenza consente ad Arduino di formulare precise indicazioni sul peso cui la barra è sottoposta.



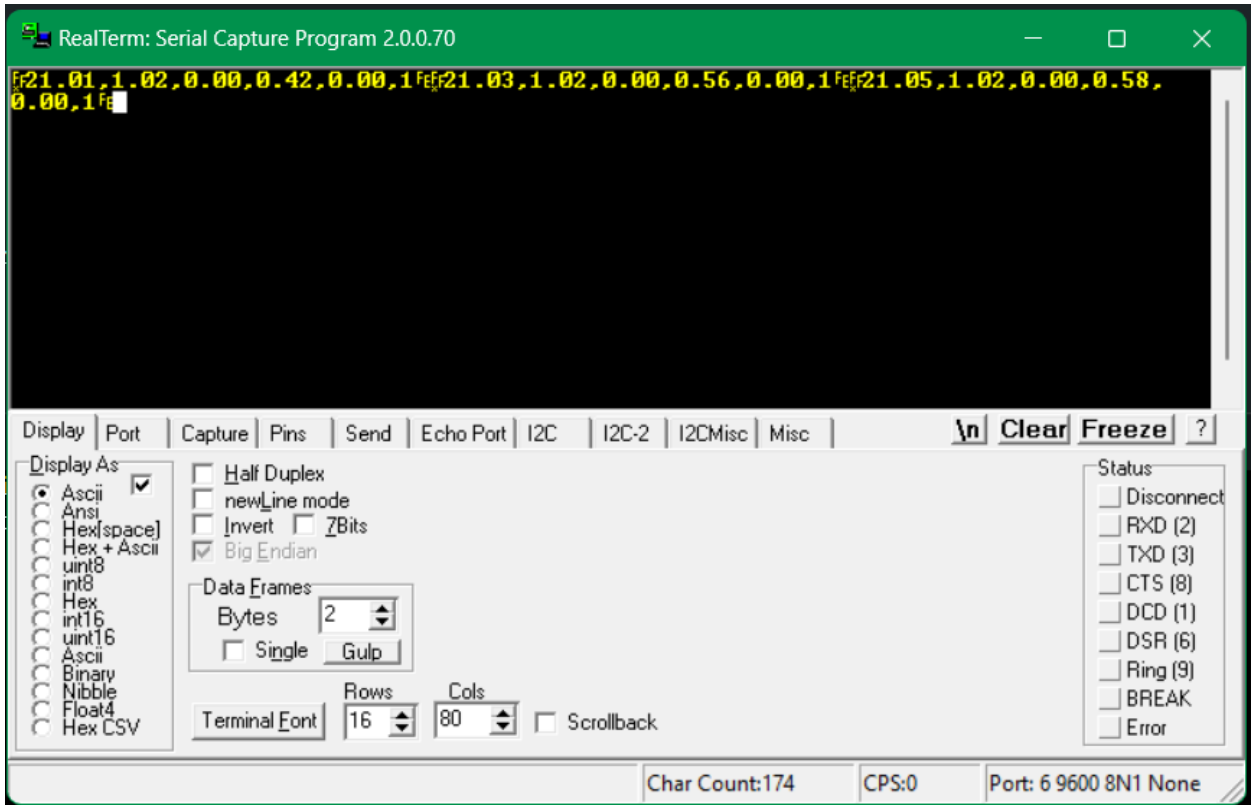
Comunicazione Micro-Bridge

Il pacchetto dati che viene continuamente inviato dal microcontrollore al bridge è così formato:

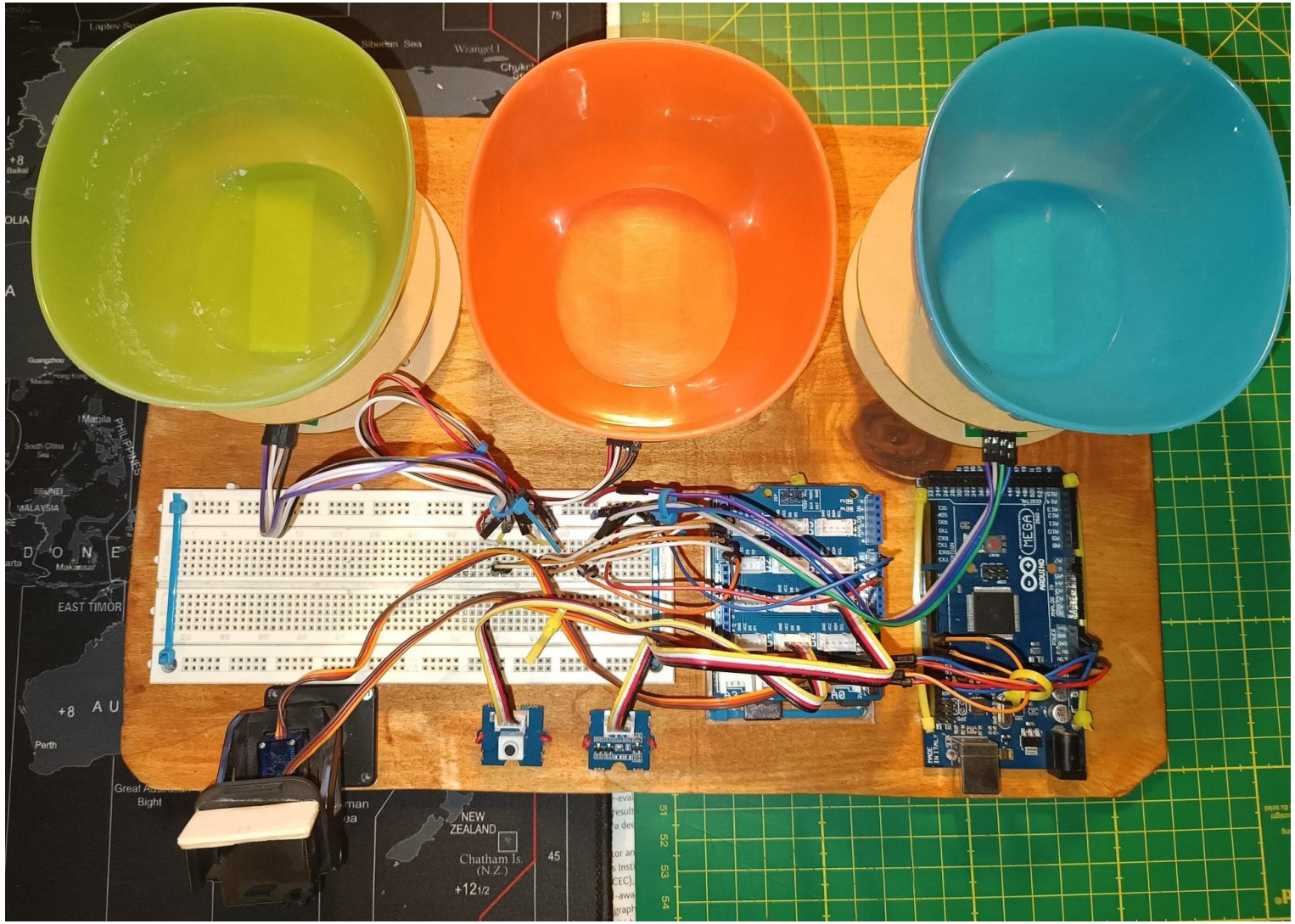
I valori dei sensori vengono spediti e ricevuti dal bridge come un' unica stringa «DATI» che viene successivamente separata dal bridge attraverso la funzione «useData» nei rispettivi valori che andranno postati su server.

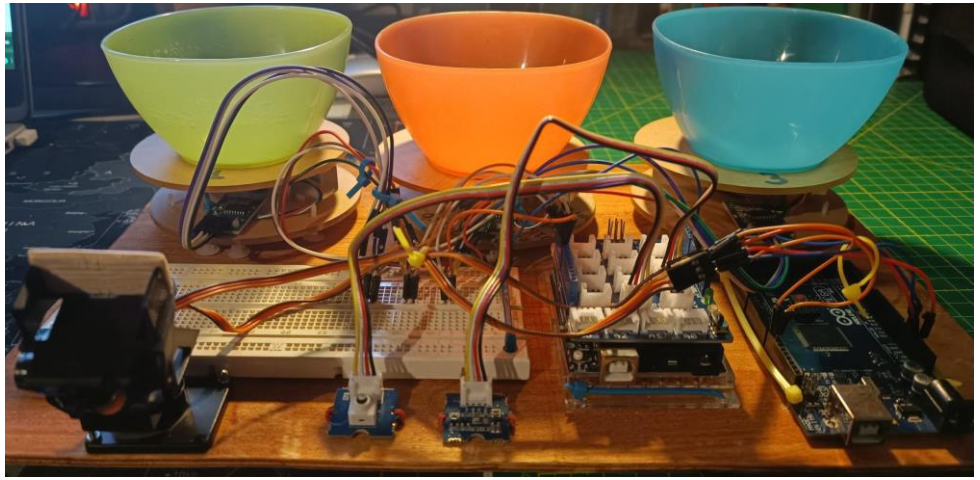


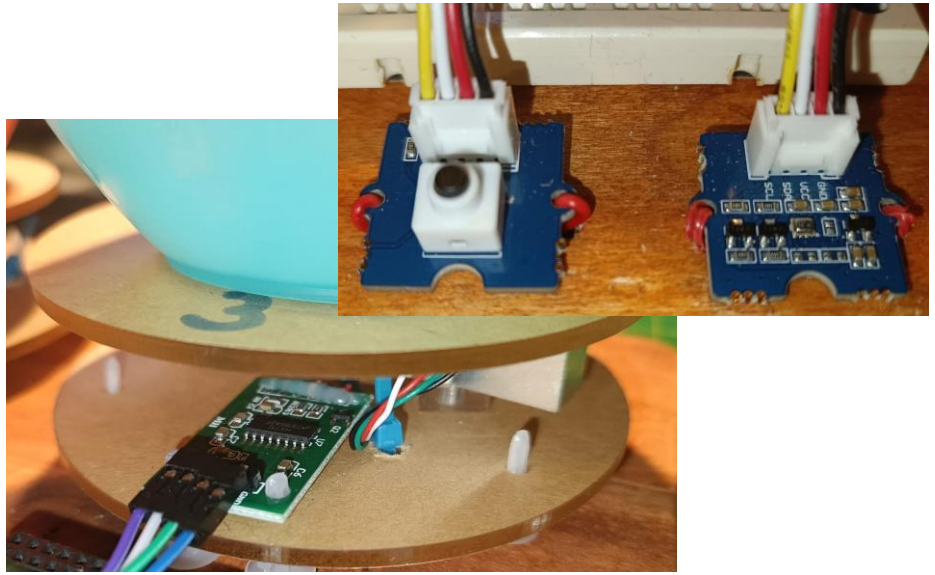
Comunicazione Micro-Bridge



Panoramica del prototipo







Specifiche del bridge

Specifiche del bridge

Possiamo dire che nel nostro prototipo sono presenti tre tipologie di bridge:

- **Il primo** è il bridge che fa da ponte tra il cloud e il microcontrollore Arduino Uno dai quali si differenzia per capacità computazionali elevate e di interconnessione, questo bridge si trova su un primo PC.
- **Il secondo** bridge fa ponte tra il cloud e il microcontrollore che controlla l'attuatore. In fase di vendita questa parte verrà integrata nel bridge precedente grazie alle comunicazioni HTTP direttamente al braccio robotico, oppure gestita separatamente come da prototipo ma da un sistema in grado di gestire le caratteristiche di un robot per potenza, voltaggio, corrente ecc.
- **Il terzo «simulatore»** bridge è quello che simula la connessione tra il cloud e le altre postazioni di lavoro fittizie che popolano il dataset sul server.

Specifiche del bridge

Il primo bridge si prende carico delle seguenti azioni:

- Ricevere dal microcontrollore gli aggiornamenti relativi ai dati della postazione di lavoro.
- Controllare che la propria postazione non sia già in fase operativa, altrimenti aggiornare, tramite post http, il digital twin dell'intero sistema con i nuovi dati che serviranno all'algoritmo decisionale.
- Se la propria postazione robot è in fase operativa, riprende a caricare i dati sul server una volta che il bottone viene premuto.

Specifiche del bridge

Il **secondo bridge** controlla la transizione della colonna “working status“ da False a True sul digital twin, sinonimo di un incarico di lavorazione da parte dell’algoritmo decisionale.

Nel caso in cui venga richiesta una lavorazione al robot di riferimento, viene inviato al microcontrollore un segnale che indicherà al Cobot di quale task eseguire.

Specifiche del bridge

Il terzo «simulatore» bridge si prende carico delle seguenti azioni:

- Popola il Digital Twin sul server con i dati relativi alle postazioni robot inserite nel sistema, simulando anche la presenza di robot che hanno valori al di fuori dal range di lavoro abilitante.
- Successivamente, se viene richiesta una lavorazione ad uno dei robot fittizi, va ad aggiorna i valori di stoccaggio dei materiali, come se una vera lavorazione fosse avvenuta.
- Quando una delle postazioni robot è in fase di lavorazione, in modo casuale aspetta il termine delle operazioni per un certo periodo di tempo, prima di indicare la fine delle operazioni sul server con una post.

Specifiche del server

Server
Specifiche

Per la creazione del server HTTP si è fatto uso del framework ***Flask***. Questo server si occupa di offrire molteplici pannelli di controllo, da cui è possibile:

- Controllare il flusso dati che viene condiviso, proveniente dai diversi Cobot inseriti nel sistema di controllo R.A.M.S.
- Controllare una parte/gruppo di Cobot.
- Decidere a quali Cobot assegnare le lavorazioni dei vari componenti.

Server
Specifiche

IDEATIVA SPIN-OFF OF UNIMORE

IoT Project

Robot Digital Twin

Id	Robot	Timestamp (YYYY/MM/DD HH/MM/SS)	Temperature (°C)	Pressure (bar)	Weight_1 (g)	Weight_2 (g)	Weight_3 (g)	Enable	Working_status	Lavorazione
23679	1	2023-02-17 17:53:47.273274	20.14	1.03	0	0.26	0.24	True	False	
24030	2	2023-02-20 11:14:13.077675	29.28	0.97	39.77	12.19	22.86	True	False	
24032	3	2023-02-20 11:14:15.167168	22.78	0.97	18.64	28.15	39.19	True	False	
24026	4	2023-02-20 11:14:08.928818	14.06	0.95	36.84	29.14	23.09	True	False	
24027	5	2023-02-20 11:14:09.989025	26.98	1.06	23.7	28.02	30.83	True	False	
24031	6	2023-02-20 11:14:14.130390	13.46	0.97	26.59	13.71	13.78	True	False	
24029	7	2023-02-20 11:14:12.045597	10.09	0.99	12.79	39.44	22.95	True	False	
24015	8	2023-02-20 11:13:57.349184	13.96	0.88	11.19	32.59	10.36	True	False	
24007	9	2023-02-20 11:13:49.001196	16.5	1.02	13.61	11.24	11.67	True	False	
24019	10	2023-02-20 11:14:01.604313	13.67	0.86	23.49	36.17	39.33	True	False	

http://101.58.103.109/lista

Server
API

Anche il server HTTP offre delle API pubbliche, che saranno sicuramente ampliate dopo la fase prototipale. Al momento gli endpoint offerti sono:

- **/** : pagina di Home.
- **/lista_completa** : per avere informazioni su tutti i dati che vengono condivisi online.
- **/lista** : visualizza informazioni relative ai robot con ID Cobot da 1 a 10.
- **/robot{ID_Robot}** : permette di visualizzare solo i dati riferiti al quel robot.

Server
API

- /addinlista/<robot>/<temperature>/<pressure>/<weight_1>/<weight_2>/<weight_3>/<enable>: per aggiungere nuove informazioni relative a quel Cobot.
- /addinlista_lavorazione/<robot>/<temperature>/<pressure>/<weight_1>/<weight_2>/<weight_3>/<enable> /<lavorazione>: per aggiungere una nuova lavorazione relativa a quel Cobot.
- /ai_algorithm : pagina che permette di poter decidere quale prodotto deve essere realizzato, in modo da suddividere le lavorazioni dei vari componenti tra i diversi Cobot.
- /temperature/<robot> : per avere il dato di temperatura del Cobot indicato.
- /pressure/<robot> : per avere il dato di pressione del Cobot indicato.

Server
API

- **/weight_1/<robot>** : per avere il dato di peso della bilancia 1 nel magazzino del Cobot indicato.
- **/weight_2/<robot>** : per avere il dato di peso della bilancia 2 nel magazzino del Cobot indicato.
- **/weight_3/<robot>** : per avere il dato di peso della bilancia 3 nel magazzino del Cobot indicato.
- **/enable/<robot>** : per vedere lo stato del Cobot indicato: *True* disponibile, *False* non disponibile.
- **/working_status/<robot>** : per vedere se il Cobot indicato sta lavorando: *True* sto lavorando, *False* non sto lavorando.
- **/lavorazione/<robot>** : per vedere quale lavorazione il Cobot sta eseguendo.

Algoritmo decisionale

Algoritmo
Di cosa devo tenere conto?

Tramite l'apposita pagina web, indicando di quale prodotto si ha necessità, è possibile suddividere le varie lavorazioni sui Cobot a disposizione. Nel prendere la decisione l' algoritmo fa le seguenti considerazioni:

- Il Cobot si trova in un range di condizioni di lavoro accettabili dal punto di vista della temperatura e pressione?
- Il Cobot sta già effettuando una lavorazione oppure è libero?
- Quali lavorazioni è in grado di fare ogni Cobot, in base alle risorse di materiali disponibili?

Algoritmo
Combinazione trovata !

R.A.M.S. è pensato per poter velocizzare la produzione grazie alla lavorazione in contemporanea di più Robot.
Tenendo in considerazione i dati di ogni Cobot, l'algoritmo cerca quali Cobot devono essere impiegati nella lavorazione dei vari *componenti*. Una volta trovata la combinazione, sul server vengono pubblicati per ogni Cobot le indicazioni di cosa devono svolgere.

Grazie per l’attenzione