**Ordini di produzione (o lotti)**

Ogni lotto sarà caratterizzato da:

**id lotto** //assegnato con un counter

**id prodotto**

**numero di pezzi**

**priorità lotto** (a eguale priorità si fa prima quello che dura meno per ridurre il tempo medio di produzione) -> scelta di un algoritmo greedy: schedulare in ordine di priorità

**tempo di lavorazione al pezzo per tipologia di macchina** (lavorazioni ordinate)

**tempo di attrezzaggio fisso per ogni macchina** (si suppone ad ogni lavorazione di dover riattrezzare la macchina)

Si ha quindi che ogni lotto è caratterizzato da più lavorazioni (da eseguire in ordine su più macchine per un certo tempo di lavorazione più il tempo di attrezzaggio).

JSON delle api che inseriscono un nuovo lotto:

{“IDProdotto”: idprod, “NumeroPezzi”: nump, “Priorita”: priorita, “DatiDiProduzione”: [{“TempoLavorazione”: tl, “TipologiaMacchina”: tm}]}

**Lavorazioni macchina**

L’output dell’algoritmo di schedulazione sarà una lista di **lavorazioni** le quali definiscono le seguenti informazioni:

**Macchina su cui la lavorazione è schedulata**

**Lotto di appartenenza della lavorazione**

**Data inizio lavorazione** (vettore, 1 per ogni fase di lavorazione) //output

**Data di fine lavorazione** (vettore, 1 per ogni fase di lavorazione) //output

//data fine lavorazione potrebbe essere la settimana successiva, in questo produciamo un lotto nuovo schedulato automaticamente per la settimana successiva solo con le componenti rimaste. Si assegna priorità più alta per gli ordini rimasti.

La lista di lavorazioni sarà accessibile tramite API con questo formato:

[{TimestampInizioLavorazione”: ti, “TimestampFineLavorazione”: tf, “CodiceMacchina”: cm, “IDLotto”: idLotto}]

**Comportamento macchina**

Ogni macchina invia periodicamente un messaggio al server contenente le seguenti info:

**Timestamp del messaggio**

**ID macchina**

**Stato della macchina:**

* + - In lavorazione
    - In attesa di lavorazione
    - Guasto
    - In attesa di materiale

**ID lotto in lavorazione**

Utilizziamo un formato di scambio dei messaggi JSON fatto nel seguente modo:

{“TimeStamp”:ts, “ID”:id, “StatoMacchina”: sm, “IDLotto”: idlotto}

Sul server ogni macchina sarà mappata da un oggetto: ogni qualvolta un’informazione è inviata al server, oltre che essere salvata in un DB generale, aggiornerà la componente sul server (questo per costruire la dashboard).

Il comportamento delle macchine sarà simulato con delle componenti MOC che gireranno su client separati dal server e comunicheranno con questo tramite API (per salvare dati sul server e per scaricare la lista delle lavorazioni). Ad ogni macchina sarà associata una probabilità di guasto e verranno gestite le altre funzioni (ad esempio terminazione di lavorazione del lotto ed inizio del successivo) sulla base delle informazioni del lotto.