Progetto di esame

Sistemi e Applicazioni di rete

BoPa Controller

Controllo di un impianto di stampaggio della plastica

Guido Borghi – Andrea Palazzi

Caratteristiche generali del progetto

- ✔ **Ambiente di lavoro**
  BoPa Controller è un sito che implementa funzioni di un controllo industriale.
  Si è ipotizzato di voler controllare un impianto per la produzione di oggetti di plastica; il macchinario risulta essere diviso in 3 sotto-macchinari:
  - ✔ **Silos dei polimeri**: un silos che contiene i trucioli di plastica pronti per essere fusi. I parametri da controllare sono: temperatura, carico e potenza assorbita dall'impianto di ventilazione.
  - ✔ **Estrusore**: macchinario che fonde la plastica e tramite una coclea inietta il materiale plastico fuso negli stampo; i parametri monitorati sono; temperatura e pressione del materiale plastico, potenza assorbita dal macchinario.
  - ✔ **Stampo**: macchinario che gestisce gli stampi (apertura e chiusura) fino a che il materiale plastico torna allo stato solido; come nel caso dell'estrusore i parametri da controllare sono temperatura e pressione del materiale plastico, potenza assorbita dal macchinario.

  Viene richiesto un controllo di tipo *soft real-time* in quanto l'impianto non presenta particolari criticità riguardo all'incolumità delle persone e dell'ambiente di lavoro; inoltre si suppone che sulla macchina sia già installato un controllore *embedded* che sorveglia le operazioni di start, stop e arresto critico.
  Dal punto di vista della *sicurezza* dell'applicazione non ci sono particolari requisiti in quanto il macchinario viene utilizzato in un ambiente di lavoro *constraint*, l'azienda, solo da operai qualificati; inoltre l'applicativo web lavora in locale e non è esposto alla rete internet globale.

- ✔ **Fine**
  Questo controllo in particolare è stato creato con il fine di:
  - ✔ Mostrare lo stato corrente del processo con i valori istantanei di ciascun sensore
  - ✔ Mostrare l'evoluzione storica di ciascun segnale
  - ✔ Inserire valori di soglia limite al fine di evidenziare in tempo reale degli allarmi
  - ✔ Evidenziare e tenere traccia dei superamenti della soglia mediante un *alarm log*

- ✔ **Tecnologie impiegate**
  Le tecnologie scelte per lo sviluppo sono:
  - ✔ **HTML**: per la descrizione degli elementi grafici
  - ✔ **CSS + Bootstrap**: per la gestione della grafica
  - ✔ **JSP**: per la creazione di pagine web dinamiche
  - ✔ **JFreeChart**: per la creazione dei grafici
  - ✔ **JDBC**: per le connessioni e le query al database
  - ✔ **MySQL**: come database

- ✔ **Modello di riferimento**
  Nella progettazione e successiva implementazione dell'applicazione si è cercato di seguire, per quanto possibile, il modello **MVC** (Model View Controller), per cercare di rendere più efficiente la stesura e la riusabilità del codice e la gestione generale dello sviluppo.

- ✔ **Tools di sviluppo**
  Per l'host server, per la stesura e la sincronizzazione del codice sono stati utilizzati:
  - ✔ **Tomcat Server 7.0.52**
  - ✔ **Geany**
  - ✔ **GitHub**

Scelte operate e considerazioni

- **Approccio**: l'approccio privilegiato è stato di tipo *funzionale*, il più possibile aderente alle caratteristiche delle macchine da controllare e dei requisiti richiesti.
  Sono state quindi create *ad hoc* delle pagine JSP per ognuno dei sotto-macchinari che gestiscono in maniera separata la lettura dei dati specifici, la loro stampa a monitor, il controllo con i valori di soglia ecc.

- **Scalabilità**: sono stati inseriti a livello di progetto alcuni accorgimenti per rendere il software maggiormente scalabile in presenza di un utilizzo prolungato ininterrotto e quindi a fronte di una grande quantità di dati da gestire provenienti dai sensori.
  Innanzitutto si è cercato di stimare il numero di valori che il database deve memorizzare nel peggior caso (funzionamento continuo della macchina) e nel periodo di riferimento, il mese, in quanto superata tale soglia le tabelle di svuotano automaticamente.
  Si è ipotizzato che ogni sotto-macchinario produca 3 valori ogni 3 secondi; essendo 3 i sotto-macchinari si parla di 3x3x20=180 valori complessivi generati ogni minuto. Partendo da questo dato si calcola che ci siano 10800 valori all'ora, 259200 valori al giorno e 7776000 valori al mese.
  Essendo ciascun valore memorizzato in una variabile di tipo integer che, da quanto si apprende dalla documentazione di MySQL, occupa 4 byte, si può stimare un'occupazione del database di circa 233 MB, suddivisa in 3 tabelle, risultato compatibile anche con macchine e sistemi operativi datati, come si può osservare nell'immagine seguente sempre tratta dalla documentazione di MySQL:

Operating System	File-size Limit
Win32 w/ FAT/FAT32	2GB/4GB
Win32 w/ NTFS	2TB (possibly larger)
Linux 2.2-Intel 32-bit	2GB (LFS: 4GB)
Linux 2.4+	(using ext3 file system) 4TB
Solaris 9/10	16TB
Mac OS X w/ HFS+	2TB

  E' stato scelto di utilizzare il tipo INT per lasciare possibilità di modificare le unità di misura di riferimento, in quanto eventuali cambiamenti ne farebbero oscillare fortemente i valori (ad es. bar – pascal, pound/foot^2).

  Si escludono da tale calcolo le tabelle delle soglie, in quanto di dimensione trascurabile, e la tabella degli allarmi in quanto considerati eventi eccezionali e quindi poco numerosi.

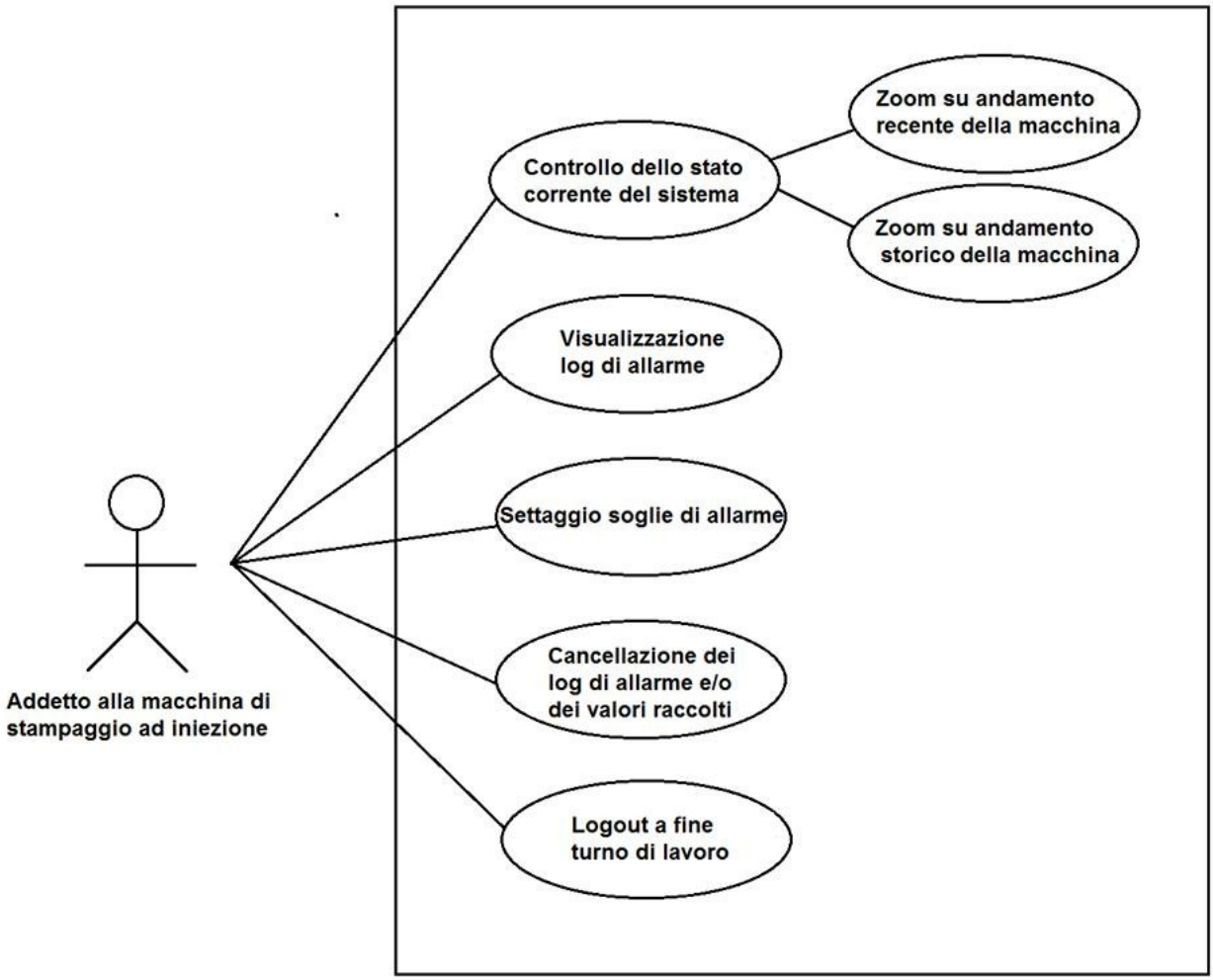
  (https://dev.mysql.com/doc/refman/5.0/en/integer-types.html)

- **Simulazione**: a causa della particolarità del dominio della presente applicazione web, si è dovuto ricorrere ad un simulatore per verificare l'efficacia del software. Non essendo tuttavia tra gli obiettivi di questo progetto la realizzazione di una simulazione realistica e la conoscenza concreta di ciò che avviene in una macchina di stampaggio a iniezione durante il suo funzionamento, si è scelto di limitare all'indispensabile la complessità di questa parte dello sviluppo.

  Di fatto, se la simulazione è attiva, ogni *n* secondi vengono inseriti valori pseudo-casuali nelle tabelle che modellano le parti della macchina di stampaggio: in un contesto reale, tali valori dovrebbero provenire dalla rete di sensori installata a bordo del macchinario.

Struttura del software

Si riporta di seguito lo Use Case Diagram a descrizione delle funzionalità principali del software :



Come evidenziato dal diagramma Use Case, le funzionalità offerte dall'applicazione sono essenzialmente rivolte alla raccolta e all'analisi dei dati provenienti dalla macchina in tempo reale.

L'unico attore coinvolto è l'impiegato addetto alla macchina di stampaggio, che utilizza il software per monitorare lo stato del sistema e verificare che non ci siano allarmi in corso o passati.
Si è prestata quindi particolare attenzione a rendere visibili tutti questi dettagli da una sola schermata iniziale e delegare a schermate secondarie la visualizzazione dei particolari (grafici e storico dei valori); inoltre nello sviluppare la parte grafica si è tenuto conto di un possibile impiego in torrette con tecnologia *touchscreen*.
Tale addetto ha anche la responsabilità di stabilire quali siano le soglie superate le quali i valori ricevuti dai sensori sono da considerarsi pericolosi per il sistema.

Qualora l'addetto decida che i valori memorizzati nel database non sono più significativi per la comprensione dell'andamento attuale della macchina (es: viene cambiato l'oggetto in stampa, quindi si registrano nuovi valori di temperatura/pressione/carico nelle varie parti del sistema), tali informazioni possono essere cancellate tramite un *flush* delle tabelle del database.

A fine turno, l'impiegato effettua il logout per tornare alla schermata principale e sfruttare la modalità di risparmio energetico.

Struttura del database

Lo schema E/R per la descrizione del database non modella particolari relazioni in quanto il database è stato pensato come pura interfaccia di raccolta dei dati che collega la macchina, responsabile della scrittura dei valori, al software di controllo, che legge i valori e memorizza altri parametri importanti come ad esempio le soglie massime.
Di seguito viene riportato il nome delle tabelle e la relativa struttura:

silos:

Campo	Tipo
temperatura	int(11)
carico	int(11)
energia	int(11)
data	timestamp

estrusione:

Colonna	Tipo
temperatura	int(11)
pressione	int(11)
energia	int(11)
data	timestamp

stampo:

Colonna	Tipo
temperatura	int(11)
pressione	int(11)
energia	int(11)
data	timestamp

soglie_silos:

Colonna	Tipo
temp_sil	int(11)
car_sil	int(11)
ene_sil	int(11)

soglie_estr:

Colonna	Tipo
temp_estr	int(11)
car_estr	int(11)
ene_estr	int(11)

soglie_sta:

Colonna	Tipo
temp_sta	int(11)
car_sta	int(11)
ene_sta	int(11)

allarmi:

Colonna	Tipo
macchina	varchar(20)
descr	varchar(30)
data	timestamp

Le tabelle *silos*, *estrusione* e *stampo* raccolgono i dati dei relativi sotto-macchinari; il campo *data* rende ogni record univoco rispetto agli altri e permette operazioni sui dati, come ad esempio la stampa dello storico dei valori; queste tabelle sono riempite di valori casuali dalla pagina JSP con funzioni di simulatore (*simulator.jsp*).

Le tabelle *soglie_sil, soglie_estr e soglie_sta* memorizzano i valori limite di soglia impostati manualmente dall'operatore e vengono continuamente confrontati con i valori letti dalle tabelle precedenti per rilevare eventuali allarmi.

Gli allarmi sono memorizzati nell'omonima tabella, anch'essi identificati univocamente dall'attributo data; nella colonna *macchina* viene memorizzato il nome del macchinario che ha generato l'allarme, nella colonna *descr* viene stampato il sensore che ha prodotto un valore al di sopra della soglia massima.

Modalità interazione client-server

L'attore principale lato client è il browser, in quanto l'operatore interagisce con il software utilizzando appunto il browser.

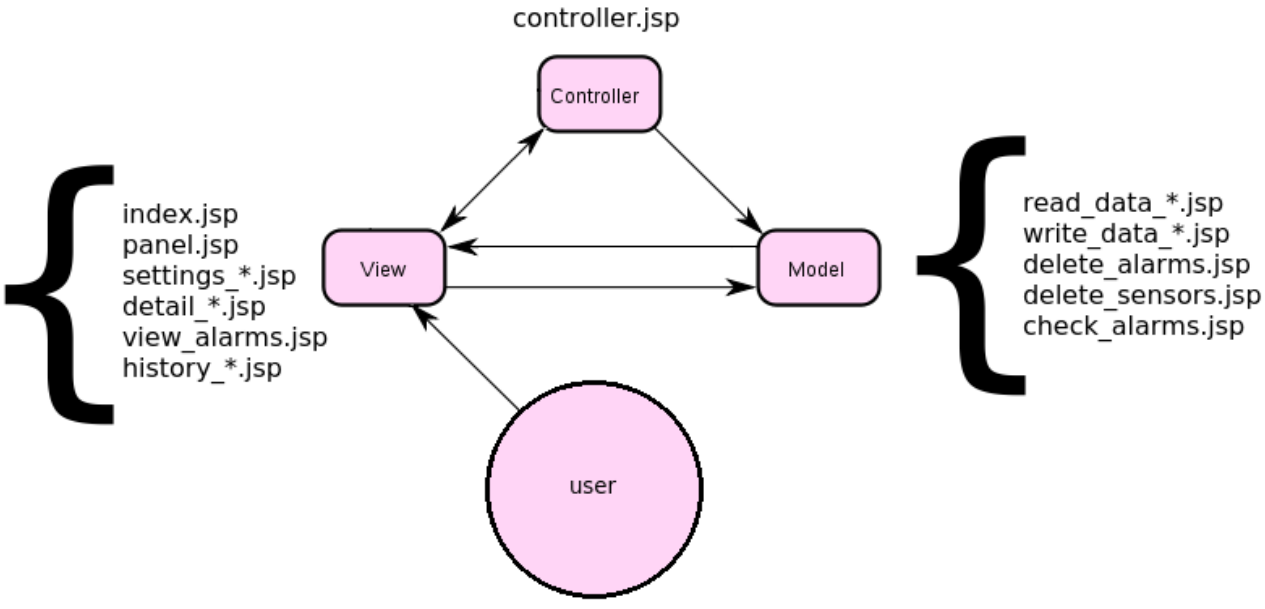
Si è cercato di mantenere per tutto lo sviluppo del progetto il modello Model View Controller, scelta che si riflette anche nell'organizzazione delle cartelle e dei file JSP del software.
Tutto il progetto è sviluppato completamente tramite la tecnologia JSP (con qualche eccezione per il discorso dei grafici e quindi le librerie JFreeChart).

Le parti presenti sono:

- **View**: si occupa di visualizzare i dati contenuti nel model e dell'interazione con l'utente.
- **Model**: esegue l'accesso al database con la conseguente scrittura o lettura dei valori.
- **Controller:** viene utilizzata per la verifica dei parametri inseriti dall'utente (ad esempio che le soglie siano valori numerici, positivi ed entro una certa dimensione) e per la gestione del Model che si comporta in maniera conseguente ai parametri trasmessi appunto dal controller.

L'utente quindi si approccia al software mediante le pagine JSP della view; alcune di queste pagine interrogano in modo diretto il Model per reperire i dati da visualizzare senza passare attraverso il Controller (del resto per queste pagine non è prevista interazione con l'utente).
Un esempio concreto è la pagina principale, *panel.jsp*, che interroga il database direttamente tramite le pagine di *read_data_*.jsp* per il reperimento dei dati dei sensori o la pagina *read_alarms.jsp* per la lettura del log degli allarmi.
Nel caso di un input da parte dell'utente (ad esempio una nuova impostazione delle soglie nelle pagine *settings_*.jsp*, la procedura per l'uscita dal programma, l'eliminazione del log degli allarmi o dei valori dei sensori tramite *delete_alarms.jsp* e *delete_sensors.jsp*...) viene invocato il Controller che, in base ai parametri passati dalla pagina chiamante, decide quale risorsa richiamare: ad esempio potrebbe passare la richiesta al Model per la modifica del database tramite le pagine *write_data_*.jsp* oppure richiamare una pagina della View per la notifica di errore.

Questa modalità di interazione è rappresentabile secondo lo schema MVC sotto riportato:



Conclusioni

L'obiettivo di questo progetto era la realizzazione di una *web application* che permettesse di raccogliere e analizzare i dati provenienti da un processo industriale, nonché di riconoscere eventuali situazioni di comportamento insolito e/o pericoloso della macchina.

Nel complesso, si pensa che tale obiettivo sia stato completamente raggiunto.
Il sistema realizzato permette infatti all'utente sia di monitorare in tempo reale lo stato del macchinario, sia di avere una visione d'insieme della storia passata dei sensori. Inoltre, sono scatenati allarmi nel momento del superamento di determinati valori di guardia: tali allarmi sono sia visualizzati nell'interfaccia principale, sia memorizzati sul database per favorire una successiva consultazione. Le suddette soglie di pericolo sono impostabili dallo stesso addetto alla macchina tramite l'applicativo web.

Si è cercato di aderire, per quanto possibile, al modello MVC. Anche se, principalmente a causa della particolarità del dominio del presente progetto, non sempre tale aderenza è perfetta, in tutte le scelte di progettazione e implementazione è stato preso da riferimento tale modello.

All'aspetto prestazionale e di scalabilità non è stata data grande rilevanza.
Si è invece cercato di privilegiare un'interfaccia grafica diretta, essenziale, con un uso generoso di grafici, in modo da massimizzare l'usabilità dell'applicazione e renderla facilmente utilizzabile anche da parte di utenti che non hanno nessun tipo di competenze informatiche. Per la stessa ragione, avendo ipotizzando che l'interfaccia con l'utente sia di tipo *touch screen*, tutte le pagine dell'applicazione sono “mono-schermata”, fruibili quindi senza utilizzo di *scroll*.