

Progetto Smartness Shelf



Progetto SmartnessShelf

Obiettivo del progetto

Realizzare un sistema software per l'ottimizzazione delle vendite all'interno di un negozio. Il sistema deve essere "distribuito" e dotato di una propria "intelligenza"

Analisi del problema

Gli esperti in vendita e marketing hanno individuato diversi parametri che legano la disposizione dei prodotti con la propensione all'acquisto.

Per il nostro progetto abbiamo selezionato, a titolo esemplificativo, 3 situazioni tipiche:

1. Merce esposta all'ingresso del negozio
2. Merce esposta all'interno del negozio
3. Merce esposta in prossima della cassa (uscita del negozio)

I criteri su cui basarsi per la disposizione della merce sugli scaffali dipende fortemente dalla situazione in cui ci troviamo, ovvero se siamo nella situazione 1, 2 o 3.

Procediamo con ordine e vediamo le tre situazioni.

SITUAZIONE 1 Scaffale all'ingresso del negozio

Gli esperti in vendita ci dicono che in questa situazione dobbiamo attirare in tutti i modi all'interno del negozio e coinvolgerlo in un'esperienza di acquisto. Ma come convinciamo il potenziale cliente ad entrare ?

Dobbiamo stimolarlo ed invogliarlo proponendogli dei prodotti in offerta.

Il potenziale cliente vede tutta una serie di prodotti scontati, con vere offerte su prodotti interessanti per cui è fortemente tentato ad entrare nel negozio.

SITUAZIONE 2 Scaffale posto all'interno del negozio

Questa situazione è diversa dalla precedente. Qui il cliente è già entrato in negozio. Molto probabilmente è entrato ed è interessato ad un prodotto in particolare o a più di uno.

Sappiamo, perché ce lo dicono gli esperti, che chi acquista il prodotto X ha una certa propensione all'acquisto del prodotto Y.

Ci sono infatti delle "regole associative" che dobbiamo conoscere per ottimizzare la nostra capacità di vendita. Tali regole d'altronde sono molto complicate, se non impossibili, da trovare mediante procedure manuali.

Fortunatamente l'informatica ci viene in aiuto e nel corso degli anni sono stati sviluppati algoritmi sofisticati (ma purtroppo con tempi di calcolo notevoli) atti a ricercare le regole associative.

L'algoritmo utilizzato nel progetto è Apriori.

SITUAZIONE 3 Scaffale posto in prossimità della cassa

Questo caso è a sua volta differente dai due casi precedenti. Qui il cliente ha già fatto i suoi acquisti principali e si trova alla cassa e magari sta facendo la fila.

Nei supermercati di tipo food alle casse si trovano sempre le caramelle, i cioccolatini ed altre cose che attirano l'attenzione dei bambini. Per un genitore in fila alla cassa è difficile resistere alle insistenze di un bambino....

Ultimamente nei supermercati food troviamo alla cassa anche le ricaricabili per i telefoni . Perché alla cassa ? Perché in un supermercato ci entriamo principalmente per fare la spesa quindi non andiamo certo a cercare le ricaricabili, però alla cassa le vediamo e allora ci viene in mente che abbiamo poco credito sulla scheda ed acquistiamo la ricaricabile.

Nei supermercati NO-FOOD alla cassa troviamo sempre prodotti a basso prezzo come ad esempio piccola cancelleria, elastici per capelli, lampadine o pile.

Sono prodotti che vanno a completare i nostri acquisti.

Controllo sulle vendite

Prima dell'avvento dei sistemi informativi la disposizione dei prodotti sugli scaffali era un'operazione piuttosto approssimativa.

Il classico processo di vendita/approvvvigionamento di un negozio era il seguente:

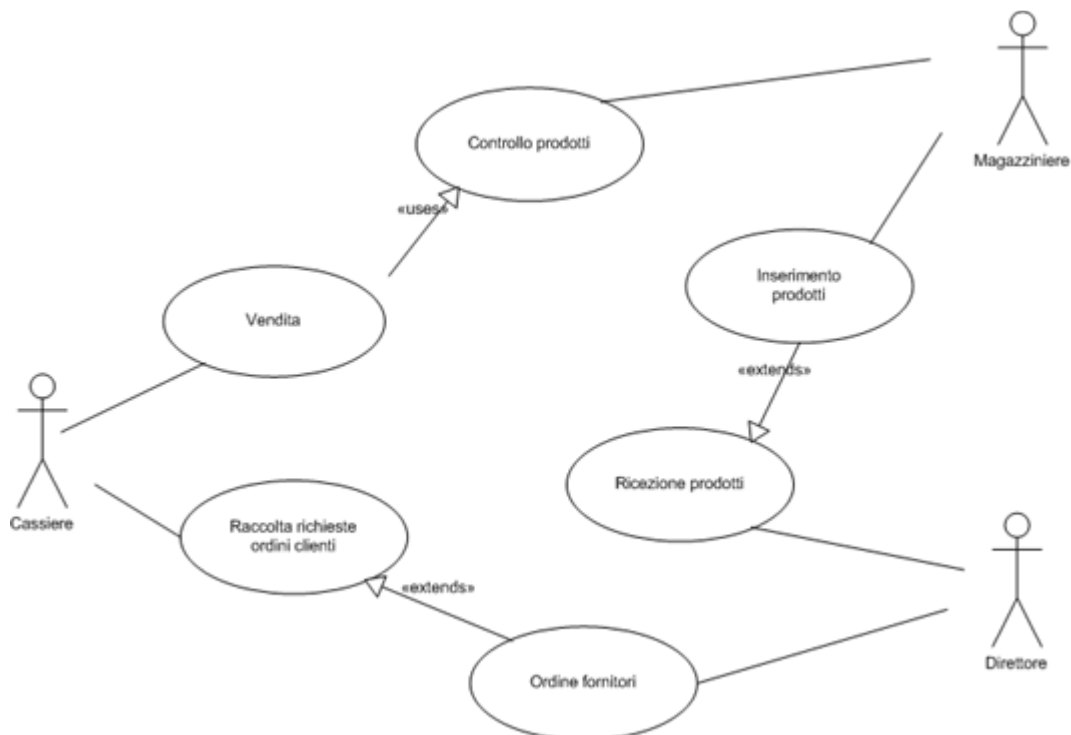


Figura 1 Processo Classico di vendita/approvvvigionamento

In pratica tutto dipendeva dall'abilità e dall'esperienza del direttore e del magazziniere. Non vi era un metodo scientifico, induttivo o deduttivo, ma solo ipotesi. Inoltre non c'era un supporto informatico atto ad aiutare il personale addetto.

Quello che ci proponiamo di sviluppare è un sistema "intelligente" che ci permetta di disporre la merce sugli scaffali in maniera scientifica ed automatica, liberandoci da possibili errori umani ed diminuendo in maniera considerevoli i costi di gestione:

- Eliminazione di prodotti difficilmente vendibili
- Abbattimento delle scorte in magazzino grazie ad un'accorta politica degli approvvigionamenti supportata da un software in grado di "predire" le quantità di merce venduta grazie a calcoli basati sullo storico delle vendite e su parametri come festività e altro.
- Diminuzione dei costi del personale in quanto non c'è più bisogno di un controllo costante della merce sugli scaffali.

Attualmente è possibile dotare gli scaffali di display in grado di visualizzare il codice prodotto, la descrizione, la marca, il prezzo e la quantità giacente in magazzino o sullo scaffale. Inoltre gli scaffali possono essere dotati di rilevatori in grado di "sentire" la presenza del prodotto.

Collegando il tutto al sistema informativo siamo in grado di sapere tutto quello che accade allo scaffale. Inoltre ci viene segnalato in automatico quando un prodotto è esaurito o sta per esaurirsi, in questo modo il personale può provvedere velocemente al riempimento dello scaffale evitando di perdere delle possibili vendite.

Ci deve essere un controllo costante delle informazioni relative alle vendite, qui di seguito vediamo in figura un modello iterativo tra la vendita e la raccolta dati.

Vediamo come funziona: Vendiamo un prodotto e questa informazione viene salvata nel database, in pratica si salvano i dati relativi allo scontrino fiscale o alla fattura emessi. Questi dati vengono elaborati con degli algoritmi per ricavare regole associative o mediante tecniche di datawarehouse e data mining. Con i risultati ottenuti vengono elaborate nuove proposte di vendita che generano degli acquisti.



Figura 2 Processo iterativo di vendita e raccolta dati

Perché tutto questo ? Perché i gusti dei clienti cambiano, cambiano le esigenze e il cliente è più informato. Monitorando in continuazione l'andamento delle vendite ed i prodotti venduti siamo in grado di seguire il mercato senza farci trovare spiazzati. Inoltre siamo sempre in grado di proporre al cliente quello che effettivamente desidera.

PROGETTO DI SMARTNESS SHELF

Vediamo ora come progettare il nostro sistema. Chiaramente stiamo sviluppando solo una parte dei requisiti visti nella fase di analisi per problemi legati alle tempistiche. Comunque il sistema sarà progettato rispettando sia i requisiti funzionali che quelli non funzionali (espandibilità, scalabilità, robustezza, sistema distribuito).

In figura possiamo vedere la rappresentazione grafica di quella che è la nostra visione esemplificativa di un sistema di vendita.

Come si può chiaramente osservare, il sistema è completamente distribuito e interfacciato al sistema gestionale già presente in azienda.

C'è uno scambio costante di informazioni tra il magazzino e le altre entità in gioco.

Questo perché quello che cerchiamo di fare è un'azione sinergica, mirata ed ottimizzata il cui scopo finale è il miglioramento costante delle vendite.

Lo scaffale va riempito in base ai dati proveniente dai risultati algoritmici ma anche in base ai dati di disponibilità forniti dal magazzino. Il magazzino invece deve sapere qual è la situazione dei prodotti sugli scaffali ma deve conoscere anche le previsioni di vendita per poter ottimizzare gli acquisti.

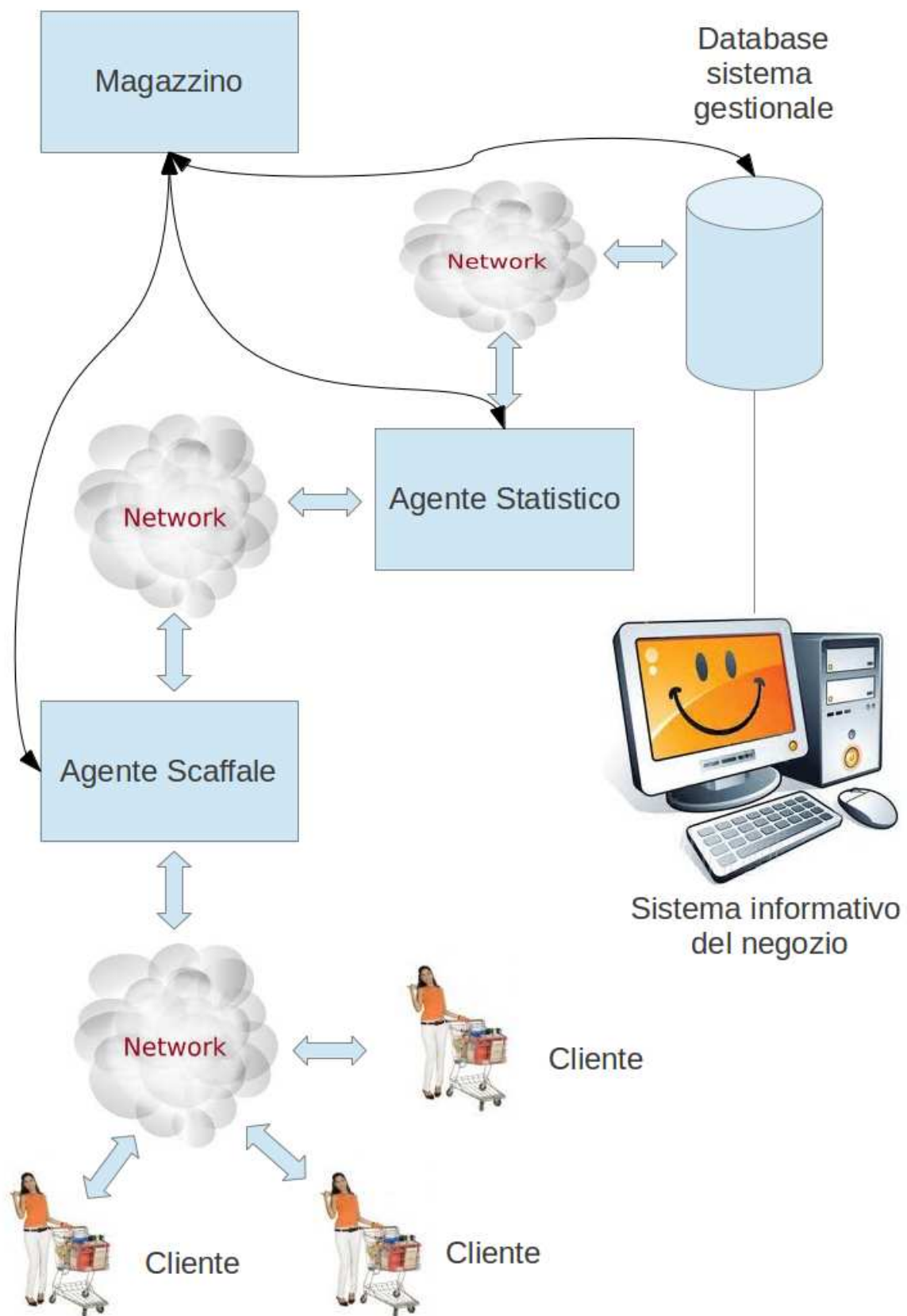


Figura 3 Rappresentazione generale del sistema

Poiché il mondo si sposta sempre più verso una realtà “MOBILE” e visto che le connessioni Internet sono diffusissime grazie anche alle capacità dei dispositivi mobili di connettersi in modalità wi-fi o 3G, si è optato per un sistema ad agenti e distribuito spazialmente.

Il fatto di avere un sistema ad agenti permette una progettazione ed una realizzazione più libera e con meno vincoli. E' una progettazione non vincolata dal linguaggio di programmazione o dalla piattaforma software/hardware.

Tra i vari agenti vi è un semplice scambio di messaggi:



Figura 4 Scambio messaggi tra agenti

REALIZZAZIONE DEL SISTEMA SOFTWARE

AGENTE STATISTICA

La realizzazione dell'Agente Statistica è stata fatta in PHP, su server Apache. La comunicazione con il resto del sistema avviene mediante scambio di messaggi in formato JSON (molto più leggero rispetto a XML). In pratica questo agente si comporta come un Web Service Server. L'agente Statistica comunica direttamente con il database aziendale. L'agente legge delle tabelle del DB precedentemente elaborate da algoritmi di Datawarehouse che in questo contesto non descriviamo.

AGENTE SCAFFALE

Per quanto riguarda l'Agente Scaffale la realizzazione è stata fatta in Java. Lo scaffale presenta due tipi di collegamento, uno verso l'agente Statistica e l'altro verso l'agente Cliente.

La parte preposta alla comunicazione con la Statistica è stata sviluppata come un Client HTTP, in questo modo la connessione con il web è molto semplice e naturale. Ha dimostrato la bontà e della flessibilità del progetto è stata sviluppata, anche se in termini grafici piuttosto grezzi, una versione in tecnologia PHP + AJAX + JSON dello scaffale al fine di dimostrare che la realizzazione degli agenti è indipendente dalla tecnologia e dai linguaggi di programmazione.

Dal lato della comunicazione con l'agente Cliente (sviluppato solo a fini di test) abbiamo utilizzato un sistema a scambio di messaggi basato sul framework JGroups. Il fatto di aver adottato questo framework ci ha permesso di tagliare in maniera considerevole i tempi di sviluppo in quanto non ci siamo dovuti preoccupare di sviluppare un sistema di messaggistica.

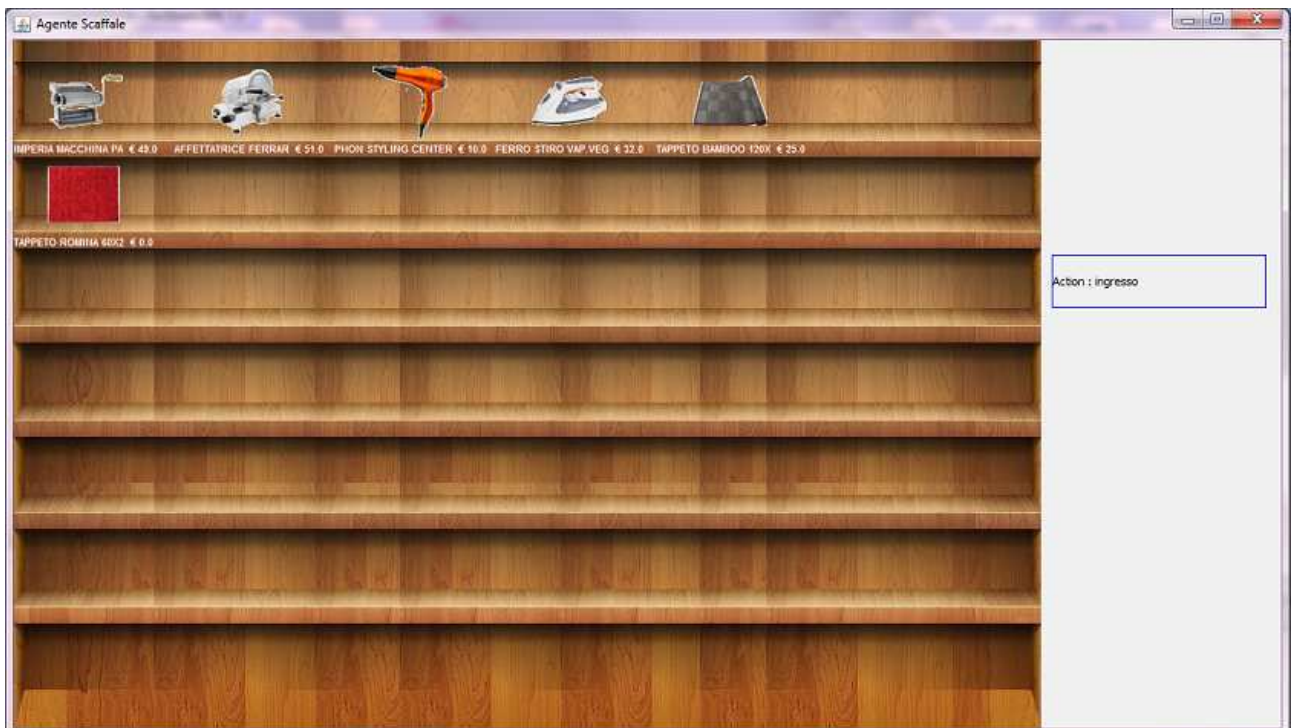


Figura 5 L'agente Scaffale in funzione

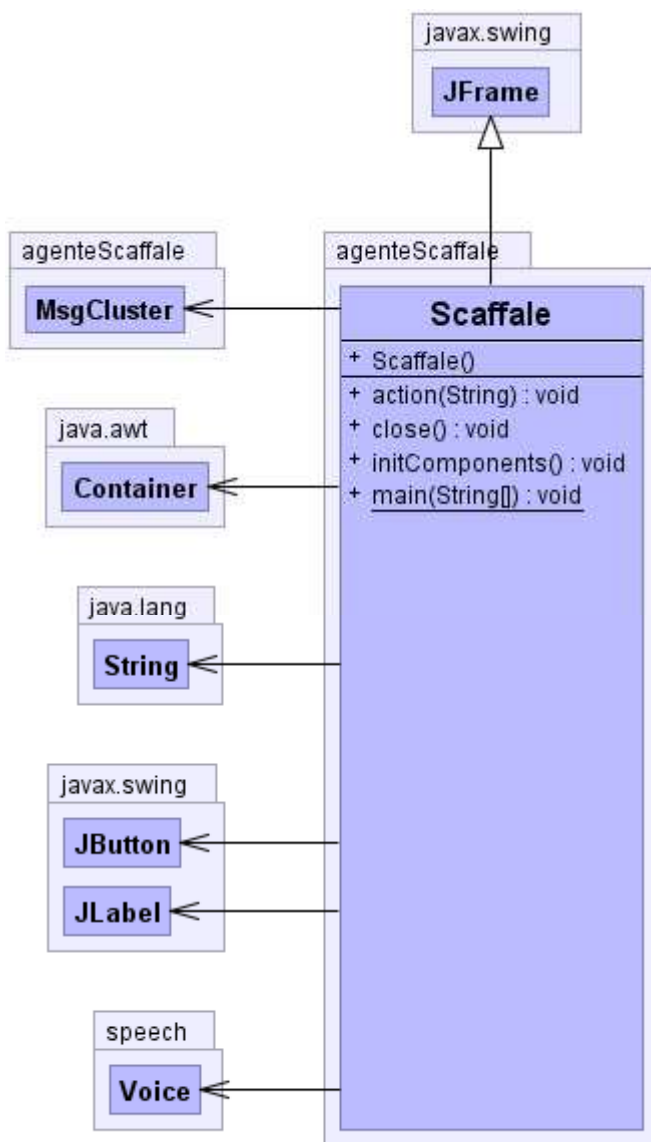


Figura 6 UML Agente Scaffold

AGENTE CLIENTE

Come si è detto l'agente Cliente (SUSAN) è stato sviluppato solo a fini di test. Anche questo agente è sviluppato in Java e dialoga con l'agente Scaffale mediante uno scambio di messaggi. Anche qui stato utilizzato il framework JGroups.



Figura 7 Package dell'agente Cliente



Figura 8 L'agente Cliente in funzione

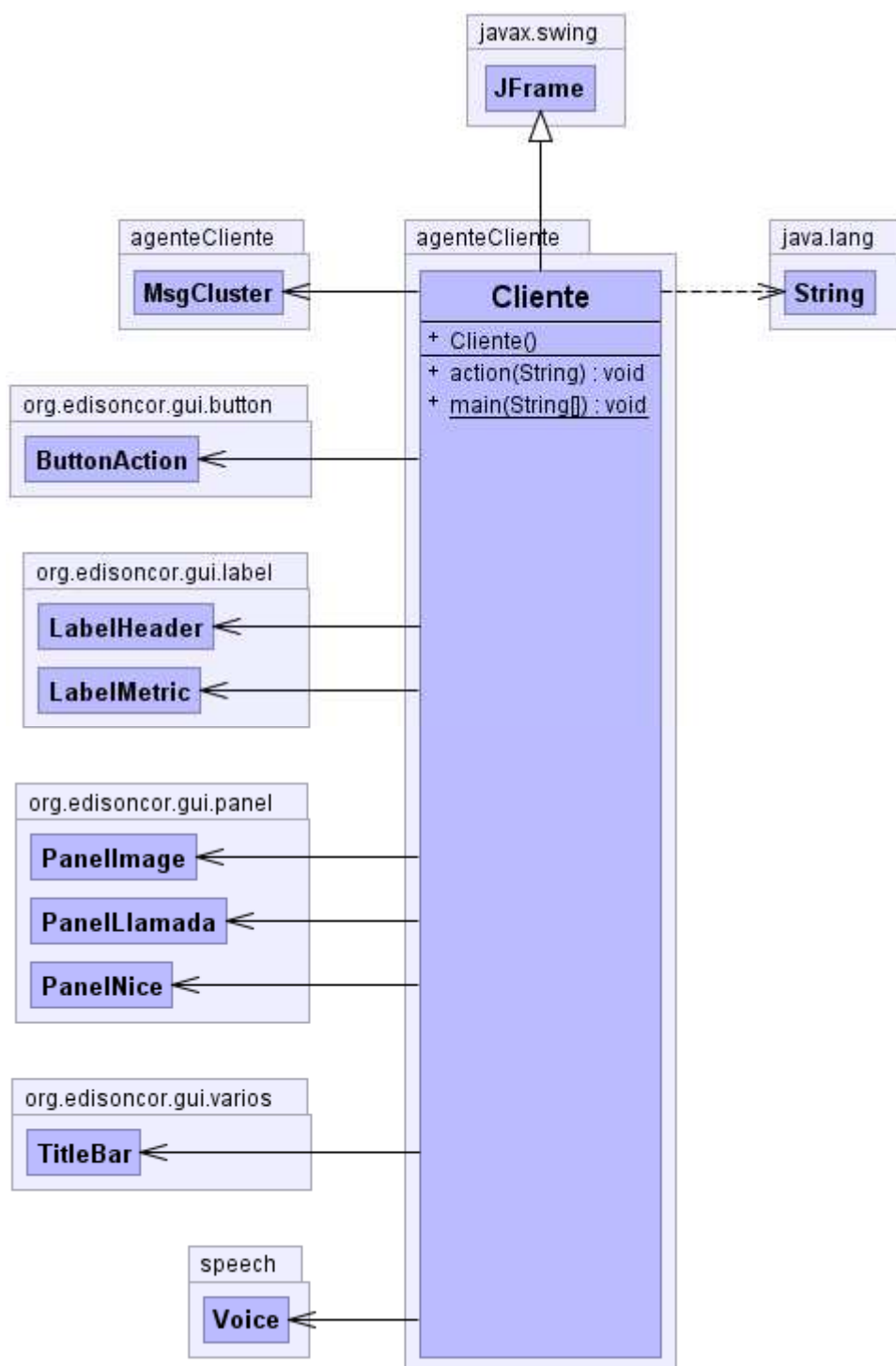


Figura 9 UML dell'agente Cliente

FRAMEWORK JGROUPS

Come accennato JGroups è l'infrastruttura scelta per la messaggistica.

Il formato dei messaggi scambiati è il seguente:



Figura 10 Formato del messaggio in JGroups

Il campo Dest rappresenta l'indirizzo del destinatario (scambio messaggi p2p). Se tale campo è NULL allora il messaggio viene inviato in Broadcast.

Src è l'indirizzo del mittente, può essere lasciato a NULL in quanto il frame work lo inizializza in automatico.

Flags contiene vari flag di configurazione. Payload contiene il messaggio vero e proprio.

Per poter abilitare il dialogo tra entità, dobbiamo creare un canale (JChannel).

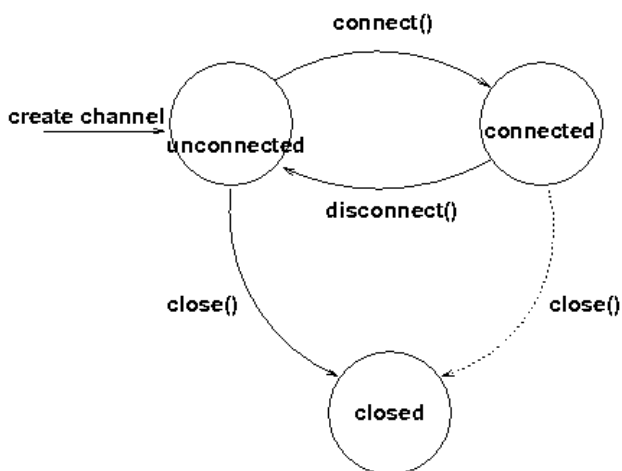


Figura 11 Funzionamento del Canale in JGroups

Come aprire il canale:

```

public class MsgCluster extends ReceiverAdapter {
    public void start() throws Exception {
        channel = new JChannel();
        channel.setReceiver(this);
        channel.connect("ShelfCluster");
        .....
        .....
    }
}

```

La classe altro non è che un thread che si occupa di ricevere i messaggi. JGroups permette di suddividere i canali in cluster, per cui due entità per dialogare devono appartenere ad un cluster. Con il metodo `connect(nomeCluster)` dico all'infrastruttura a quale cluster appartiene la mia entità.

Sviluppi futuri

Il sistema è facilmente espandibile e si presta facilmente a futuri sviluppi.

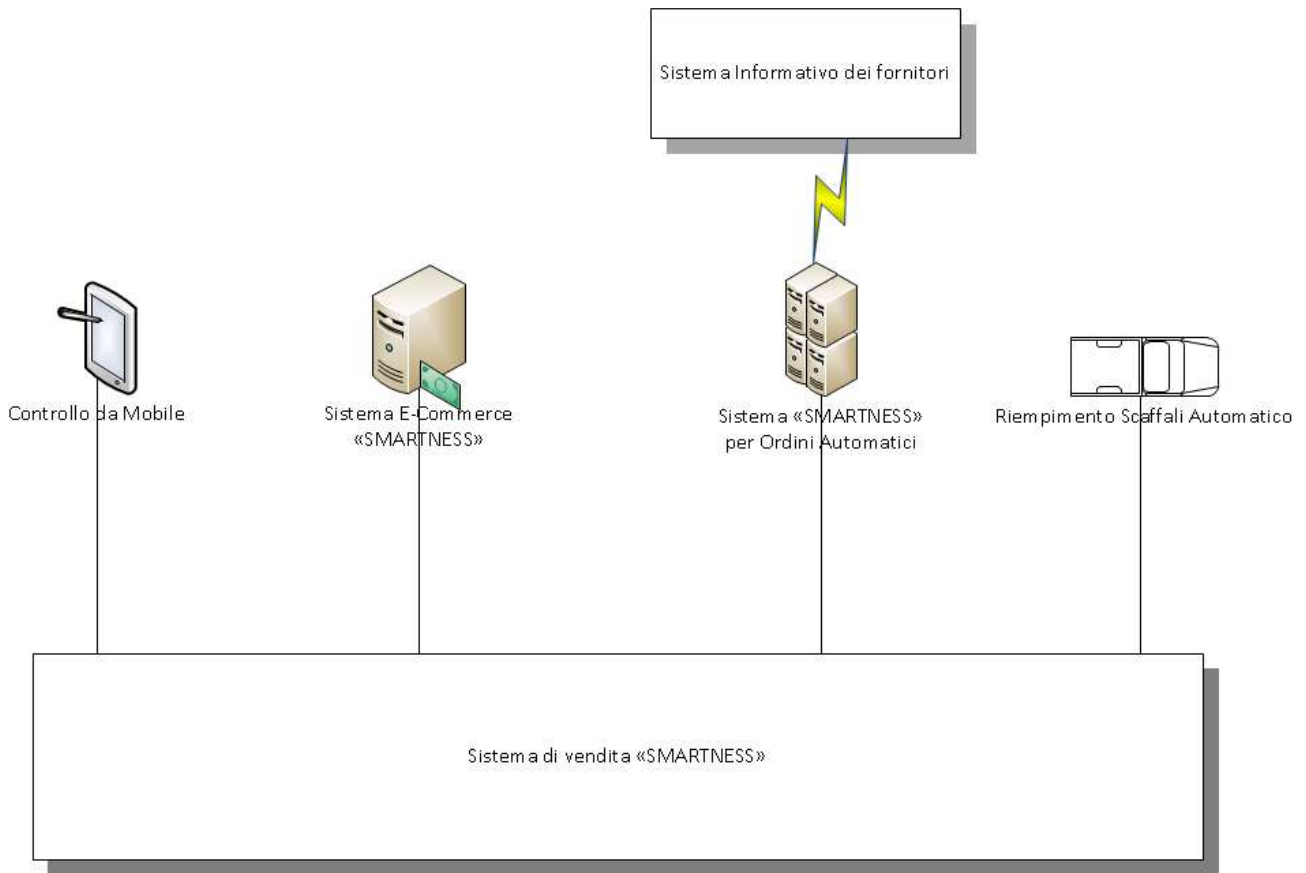


Figura 12 Sviluppi futuri del sistema SMARTNESS

Alcune idee:

- Controllo da Mobile del sistema di vendita
- Integrazione con un sistema di e-commerce "smartness"
- Implementazione di un sistema di ordini gestiti in automatico dal sistema in base alle previsioni di vendita
- Integrazione con sistemi di robot per il riempimento in automatico degli scaffali