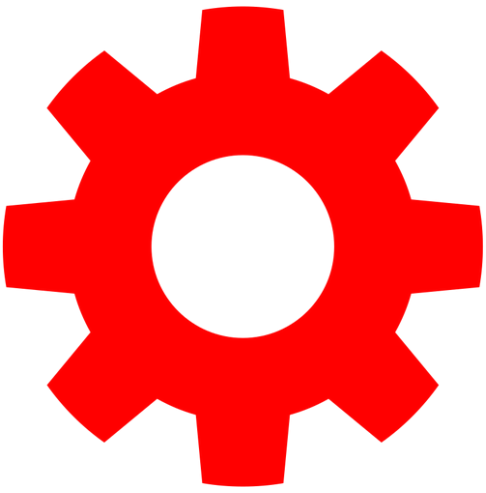
Università degli Studi di Salerno
Corso di Ingegneria del Software

SHOP-AUTORICAMBI
System Design Document - SDD
Versione 2.0



INDICE

1. INTRODUZIONE

1.1 Purpose Of The System

Si vuole realizzare un software, denominato Shop-Autoricambi, per la gestione di un sito e-commerce destinato alla vendita di pezzi di ricambio per automobili. A tal proposito, possiamo notare che un pezzo di ricambio, ossia un prodotto, può essere acquistato da uno e un solo utente e, viceversa, che un utente può acquistare più prodotti. I prodotti sono caratterizzati da un codice, dal modello, dal marchio, dal prezzo, dalla quantità e da una breve descrizione. Inoltre, un prodotto, potrebbe essere messo in offerta ed avere, quindi, un prezzo scontato.

Un utente, che si collega al sito di e-commerce mediante questo software, può registrarsi al sito e diventare un potenziale cliente.

Un utente registrato, avrà uno ed un solo indirizzo ma, viceversa, un indirizzo potrebbe corrispondere ad uno o più utenti registrati. Un indirizzo, in questo caso, è costituito da un CAP, da una via e da una città. Un utente registrato, invece, è caratterizzato da un codice fiscale che lo identifica univocamente, dal nome, dal cognome, dalla data di nascita, da un numero di telefono, da un numero di cellulare, da un'email e da una password. L'utente registrato, però, potrebbe anche essere il gestore di tale sito e svolgere funzioni diverse.

Il software che si vuole realizzare, infatti, dovrà consentire ai gestori di: autenticarsi, visualizzare lo storico degli acquisti di tutti i clienti e caricare dei nuovi prodotti da poter vendere. Il software dovrà consentire agli utenti registrati di: autenticarsi, effettuare acquisti, tenere traccia degli acquisti effettuati, tenere traccia dei prodotti scelti per un eventuale acquisto, eliminare uno o più prodotti dalla lista dei prodotti scelti per un eventuale acquisto.

Il software, inoltre, deve consentire agli utenti la ricerca dei pezzi di ricambio tramite il loro codice, la loro marca oppure il loro modello. Ogni prodotto visualizzato dal sito di e-commerce, tramite il supporto del software, avrà anche delle foto ad esso associato. In particolare, un prodotto ha una o più foto che lo descrivono ma, viceversa, una foto ha uno ed un solo prodotto a cui si riferisce.

Il software, come già descritto in precedenza, deve tenere traccia degli acquisti effettuati dai vari clienti, i quali possono avere una o più fatture, viceversa, una fattura è in possesso di uno e un solo cliente, o utente registrato. Una fattura è caratterizzata da un codice

fattura che la identifica univocamente, da una data di rilascio della stessa, da un costo totale e dal tipo di pagamento effettuato. Una fattura, di conseguenza, si riferisce ad uno o più prodotti acquistati in un certo momento. Un prodotto acquistato è caratterizzato da un codice di acquisto che lo identifica univocamente e da una descrizione.

1.2 Design Goals

- **Reliability:** Il software dovrà essere attivo 24 ore su 24. Inoltre, deve garantire la sicurezza sul tutte le operazioni effettuate sia dai gestori sia dai clienti. Nel caso in cui si verifichino dei comportamenti anomali, da parte del gestore o del cliente, verranno notificati tramite degli avvisi.

- **Performance:** Il software dovrà rispondere velocemente; le risposte dovranno essere fornite in un periodo pari a circa 1 secondo. Il numero di utenti che potranno collegarsi e acquistare prodotti contemporaneamente sarà dato dalla disponibilità del Web Server utilizzato. La latenza massima di attesa, per una risposta, non dovrà superare i 30 secondi.

- **Supportability:** Il software consisterà in un sistema client-server in cui il server sarà disponibile su ogni tipo di piattaforma e il client potrà collegarsi a tale server mediante un qualsiasi browser. Il software dovrà essere suddiviso in vari moduli per permettere una più facile modifica e aggiornabilità in futuro.

- **Implementation:** durante l'implementazione sarà utilizzata la virtual machine di java e l'ide netbeans.

- **Gestione:** il software sarà gestito da un amministratore che sarà anche il gestore del magazzino.

- **Packaging:** Il software potrà essere installato, lato server, attraverso una procedura di installazione. Il software, una volta installato dal gestore del magazzino, non sarà legato a vincoli temporali.

- **Legal:** Il software avrà licenza legata a MySql e il costo sarà legato alle percentuali di guadagno del sistema.

1.3 Riferimenti

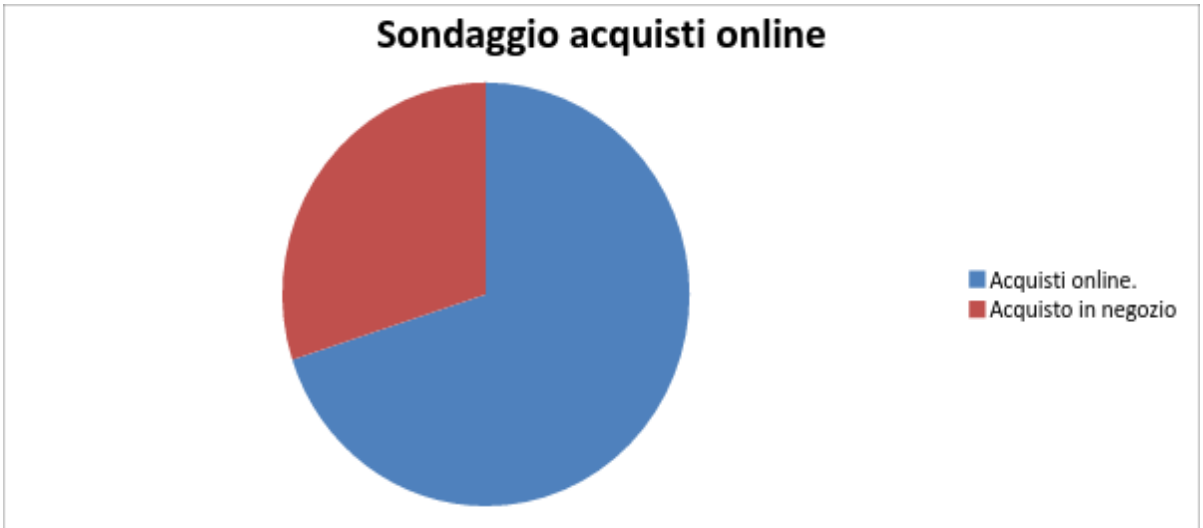
In questo documento si fa riferimento a:

- RAD_Versione_3.0
- Documentazione_Database

2. CURRENT SOFTWARE ARCHITECTURE

Non essendoci nessun sistema software da sostituire, proponiamo il risultato di un nostro recente sondaggio.

Secondo questa ricerca, è risultato che **oltre 14 milioni di italiani si sia affidato ad Internet per acquistare**, nell'ultimo anno, **pezzi di ricambio per la propria automobile**: dai semplici tergicristalli, alle lampade, alle batterie, a pezzi più complessi come freni, dischi, kit distribuzione ed altro.



La media è impressionante in quanto risulta che in un tempo inferiore a ogni 10 secondi, venga perfezionato un acquisto che riguardi l'accessoristica e la ricambistica per l'automobile, con **prevalenza al Sud Italia e Isole**. Logicamente è una media e quindi non è il dato reale visto che ciascuno di noi potrebbe acquistare più oggetti contemporaneamente, ma fa capire quanto gli italiani si affidano alla rete per la ricerca dei ricambi auto.

Quasi il 70% del campione intervistato ha addotto alla motivazione che, affidandosi alla rete, si trovano **prezzi nettamente più bassi rispetto alla rete ufficiale** o ai negozi di ricambistica; il risparmio, per la maggior parte, è quantificato in una percentuale del 20%. **La parte forte è composta da pneumatici e tergicristalli, lampadine, fusibili e batterie.**

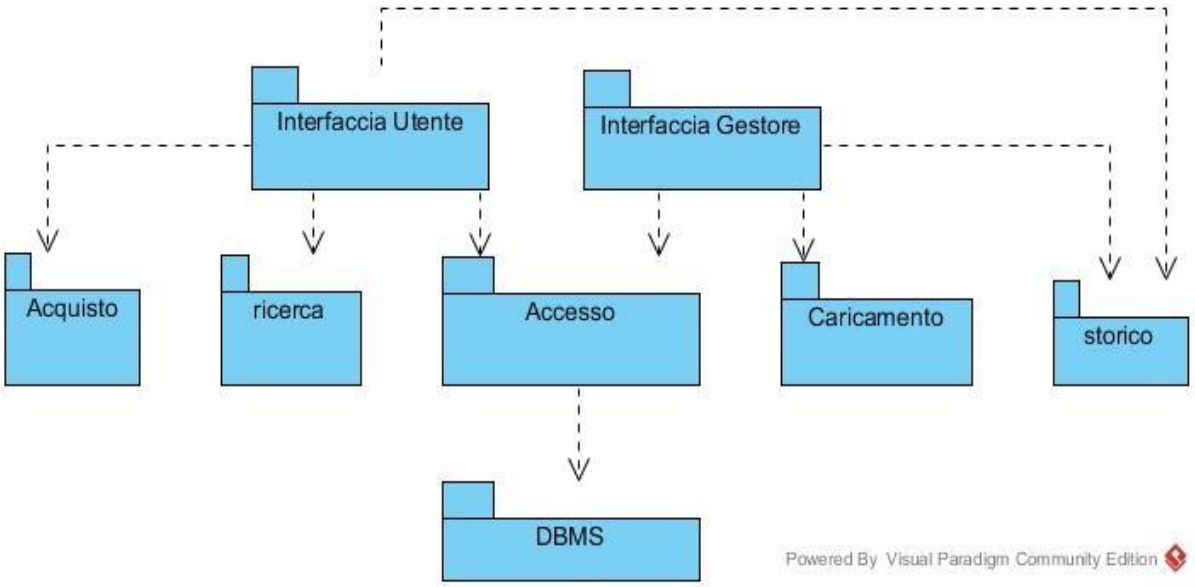
3. PROPOSED SOFTWARE ARCHITECTURE

3.1 Overview

L'architettura che è stata proposta è un'architettura software Client-Server. In questa sezione riportiamo, di seguito, in base alla tipologia di architettura scelta, le scelte effettuate partendo dalla decomposizione in sistemi fino ad arrivare alla gestione delle boundary condition.

3.2 Subsystem Decomposition

La decomposizione in sottosistemi è stata effettuata seguendo la tecnica della stratificazione (layering). La stratificazione, infatti, ha permesso l'individuazione di 3 livelli in cui classificare i vari sottosistemi in senso orizzontale. I livelli che sono stati individuati rispecchiano il modello MVC e, siccome ogni sottosistema dipende dal sottosistema immediatamente inferiore, definiamo l'architettura proposta una particolare architettura chiusa, la quale ha come obiettivo quello di favorire la manutenibilità.



I sottosistemi individuati all'interno del sistema Shop-Autoricambi, inoltre, rispecchiano, in linea di massima, le funzionalità offerte dal sistema. I sottosistemi individuati sono:

1. **InterfacciaUtente**: fornisce il servizio relativo all'implementazione dell'interfaccia degli utenti.

2. **InterfacciaGestore**: fornisce il servizio relativo all'implementazione dell'interfaccia del gestore.

3. **Accesso**: fornisce i servizi relativi all'autenticazione dell'utente.

3.1 **Login**: servizio che permette l'autenticazione al sistema.

3.2 **Registrazione**: servizio che permette la registrazione al sistema.

3.3 **Logout**: servizio che permette il logout dal sistema.

4. **Ricerca**: fornisce il servizio per la ricerca di un prodotto in base a marca, modello e codice.

5. **Acquisto**: fornisce i servizi inerenti all'acquisto di uno o più prodotti.

5.1 **Aggiungi Al Carrello**: servizio che permette di aggiungere un prodotto al carrello.

5.2 **Acquisto Prodotto:** servizio che permette di procedere con l'acquisto dei prodotti presenti nel carrello.

5.3 **Elimina Prodotto**: servizio che permette di eliminare un prodotto dal carrello.

6. **Caricamento**: fornisce il servizio per il caricamento di un prodotto da parte di un gestore.

7. **Storico:** fornisce i servizi inerenti alla visualizzazione delle operazioni effettuate.

7.1 **Storico Cliente:** servizio che permette ad un utente di visualizzare lo storico inerente ai prodotti acquistati fino a quel momento.

7.2 **Storico Dei Clienti**: servizio che permette ad un gestore di visualizzare gli storici, dei prodotti acquistati fino a quel momento, di tutti gli utenti.

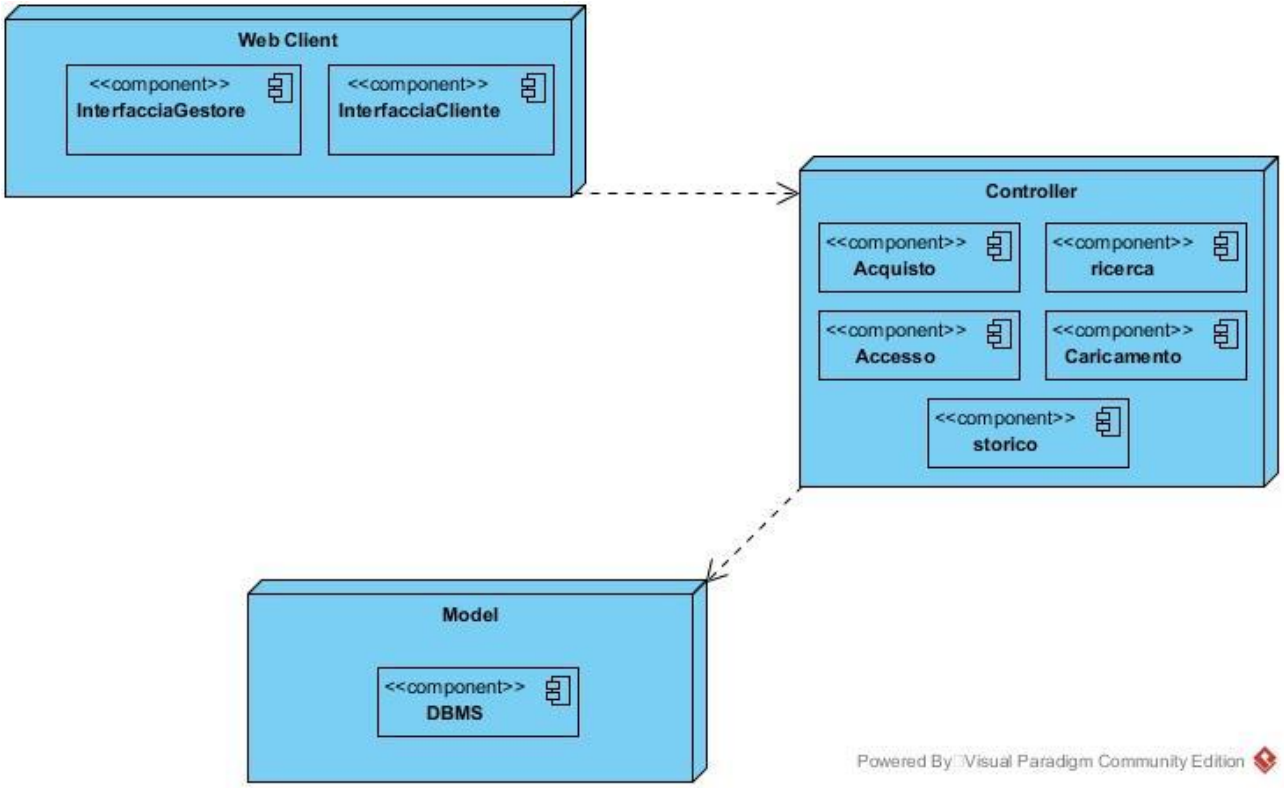
8. **DBMS**: fornisce il servizio che si occupa di interfacciare le richieste effettuate, dai vari sottosistemi, al database.

9. **Server**: fornisce i servizi per permette il funzionamento e/o la comunicazione di tutti gli altri sottosistemi.

3.3 Hardware/Software Mapping

In questa sezione è riportato il Deployment Diagram del sistema Shop-Autoricambi.

Tale diagramma ha lo scopo di visualizzare le varie dipendenze a Runtime.



3.4 Persistance Data Manager

Il sistema shop-autoricambi si avvale dell'uso di un database di tipo relazionale. Questa scelta è stata ponderata data la sicurezza offerta da un DBMS di ultima generazione, assieme ad una maggiore affidabilità e garanzia di coerenza e facilità di gestione, nonché

dalla velocità di accesso e trasmissione dei dati.

Un database di tipo relazionale, inoltre, permette la gestione e la memorizzazione permanente di un grosso insieme di dati che devono e/o possono essere acceduti da utenti e applicazioni diverse a una granularità più fine.

Ovviamente, utilizzare un database relazionale significa avere a disposizione circa il triplo dello spazio di memorizzazione richiesto per il corrispondente insieme di dati.

Altri dettagli, inerenti alla schematizzazione e all'implementazione, sono riportati nel seguente documento: **Documentazione_Database**.

3.5 Global Software Control

Il sistema Shop-Autoricambi prevede un meccanismo, per la gestione del flusso di controllo, di tipo esplicito. In particolare i controlli saranno gestiti dagli eventi (event-driven). Il controllo, infatti, risiederà in un dispatcher il quale richiamerà, a sua volta, le funzioni o funzionalità del sistema.

3.6 Boundary Conditions

Questa sezione definisce le boundary conditions che descrivono l'avvio, l'arresto e i malfunzionamenti del sistema:

- Avvio: avvio (startup) del server tomcat tramite il file startup.bat, presente nella cartella bin di tomcat.
- Arresto: chiusura (shutdown) del server tomcat tramite il file shutdown.bat, presente nella cartella bin di tomcat.
- Malfunzionamenti: in caso di malfunzionamenti del server, dovuti ad errori dello stesso o ad errori dovuti alla connessione, verrà visualizzata una pagina di errore.