**System Design Document Template**

**Purpose**

System design is documented in the System Design Document (SDD). It describes design goals set by the project, subsystem decomposition (with UML class diagrams), hardware/software mapping (with UML deployment diagrams), data management, access control, control flow mechanisms, and boundary conditions. The SDD is used to define interfaces between teams of developers and serve as a reference when architecture-level decisions need to be revisited.

**Audience**

The audience for the SDD includes the project management, the system architects (i.e., the developers who participate in the system design), and the developers who design and implement each subsystem.

**Template**

|  |  |
| --- | --- |
| **Outline** | **Description** |
| **1. Introduction** 1.1 Purpose of the system 1.2 Design goals 1.3 Definitions, acronyms, and abbreviations 1.4 References 1.5 Overview | *The purpose of the Introduction is to provide a brief overview of the software architecture and the design goals. It also provides references to other documents and traceability information (e.g., related requirements analysis document, references to existing systems, constraints impacting the software architecture).* |
|  | |
| **2. Current software architecture** | *The second section describes the architecture of the system being replaced. If there is no previous system, this section can be replaced by a survey of current architectures for similar systems. The purpose of this section is to make explicit the background information that system architects used, their assumptions, and common issues the new system will address.* |
|  | |
| **3. Proposed software architecture** | *The third section documents the system design model of the new system.* |
|  | |
| **3.1 Overview** | *The overview presents a birds-eye view of the software architecture and briefly describes the assignment of functionality to each subsystem.* |
|  | |
| **3.2 Subsystem decomposition** | *Subsystem decomposition describes the decomposition into subsystems and the responsibilities of each. This is the main product of system design.* |
|  | |
| **3.3 Hardware/software mapping** | *Hardware/software mapping describes how subsystems are assigned to hardware and off-the-shelf components. It also lists the issues introduced by multiple nodes and software reuse.* |
|  | |
| **3.4 Persistent data management** | *Persistent data management describes the persistent data stored by the system and the data management infrastructure required for it. This section typically includes the description of data schemes, the selection of a database, and the description of the encapsulation of the database.* |
|  | |
| **3.5 Access control and security** | *Access control and security describes the user model of the system in terms of anaccess matrix. This section also describes security issues, such as the selection of an authentication mechanism, the use of encryption, and the management of keys.* |
|  | |
| **3.6 Global software control** | *Global software control describes how the global software control is implemented. In particular, this section should describe how requests are initiated and how subsystems synchronize. This section should list and address synchronization and concurrency issues.* |
|  | |
| **3.7 Boundary conditions** | *Boundary conditions describes the start-up, shutdown, and error behavior of the system. (If new use cases are discovered for system administration, these should be included in the requirements analysis document, not in this section.)* |
|  | |
| **4. Subsystem services Glossary** | *The fourth section, Subsystem services, describes the services provided by each subsystem in terms of operations. Although this section is usually empty or incomplete in the first versions of the SDD, this section serves as a reference for teams for the boundaries between their subsystems. The interface of each subsystem is derived from this section and detailed in the Object Design Document.* |

Progetto di Ingegneria del software

Anno accademico 2018/2019



*Carcheck*



|  |  |  |
| --- | --- | --- |
| **Tabella Componenti** | | |
| **Capriglione** | **Francesco** |  |
| **D’Auria** | **Aldo** |  |
| **De Falco** | **Daniele** |  |
| **Iacovazzo** | **Giovanni** | 051214774 |

**Indice**

**1. Introduzione**

1.1 Scopo del sistema

[1.2 Obiettivi di design](file:///C:\Users\Amministratore\Desktop\Rub\3.%20SystemDesignDocument_Electroshock%20andrea%20ia.docx#_1zvewshm996z)

[1.2.1 Criteri di Performance](file:///C:\Users\Amministratore\Desktop\Rub\3.%20SystemDesignDocument_Electroshock%20andrea%20ia.docx#_79dbl38zbw8u)

[1.2.2 Criteri di Affidabilità](file:///C:\Users\Amministratore\Desktop\Rub\3.%20SystemDesignDocument_Electroshock%20andrea%20ia.docx#_4v4owpt8aogw)

[1.2.3 Criteri di Manutenzione](file:///C:\Users\Amministratore\Desktop\Rub\3.%20SystemDesignDocument_Electroshock%20andrea%20ia.docx#_1lqw9jlghad6)

[1.2.4 Criteri per l’Utente Finale](file:///C:\Users\Amministratore\Desktop\Rub\3.%20SystemDesignDocument_Electroshock%20andrea%20ia.docx#_lylm389f7szv)

[1.3 Definizioni, acronimi e abbreviazioni](file:///C:\Users\Amministratore\Desktop\Rub\3.%20SystemDesignDocument_Electroshock%20andrea%20ia.docx#_hisql3v3lcec)

1.4 Riferimenti

1.5 Panoramica

**2. Architettura software corrente**

**3. Architettura software proposta**

3.1 Panoramica

3.2 Decomposizione in sottosistemi

3.3 Mappatura hardware/software

3.4 Gestione dei dati persistenti

3.5 Controllo degli accessi e sicurezza

3.6 Controllo globale del software

3.7 Boundary conditions

**4. Servizi dei sottosistemi**

4.1 Gestione Utente

4.2 Gestione Annunci

4.3 Gestione Notifiche e Messaggi

4.4 Gestione Feedback

4.5 Gestione Categorie

**5. Glossario**

**1. Introduzione**

**1.1.Scopo del sistema**

Gli obiettivi del sistema sono quelli di definire una piattaforma in grado di gestire dati relativi a diverse tipologie di veicoli e ad una serie di autofficine autorizzate e aderenti al progetto CarCheck.

Un utente grazie al sistema può controllare in maniera rapida i dettagli del proprio veicolo o di qualsiasi altro veicolo interessato semplicemente conoscendo la targa. Le officine, invece, utilizzeranno il sistema per gestire con maggiore efficienza le revisioni effettuate e l’elenco dei veicoli su cui abbiano effettuato un lavoro.

Un’officina entrerà a far parte di CarCheck tramite una semplice richiesta ( da effettuare sulla piattaforma tramite l’apposito link ) e un amministratore deciderà se accettare o rifiutare tale candidatura. Infine il software si propone di essere funzionale su dispositivi di natura diversa, come ad esempio smartphone, tablet, pc desktop ecc.

**1.2.Obiettivi di design**

Il sistema dovrà guidare l’utente nell’esecuzione dei task desiderati.

Verranno utilizzati meccanismi per verificare la correttezza degli input inseriti e notifiche mostrate agli utenti in caso di successo o fallimento dell’inserimento di tali dati.

.

Parte degli aspetti che il sistema avrà sono :

1.2.1Criteri di Performance

|  |  |
| --- | --- |
| Tempo di risposta | CarCheck dovrebbe assicurare un tempo di risposta alle richieste dell’utente non superiore a 10 secondi. Tale tempo potrebbe essere maggiore in base a fattori esterni al sistema ( connessione di rete, situazioni di overloading dovute ad un numero “elevato” di utenti, ecc). |
| Memoria | Essendo un applicativo web esso risiederà su un Web Server.  Al momento non è possibile stimare la quantità di memoria che la piattaforma utilizzerà. La dimensione sarà dettata dal numero di veicoli presenti nel database. Da un calcolo approssimativo, tale dimensione “dovrebbe” essere sui 10/15 Gb effettuando una serie di supposizioni e semplificazioni. |

1.2.2 Criteri di Affidabilità

|  |  |
| --- | --- |
| Robustezza | In caso di input non validi, il sistema notificherà l’utente e richiedendo la nuova immissione di questi ultimi. |
| Sicurezza | L’accesso al sistema è controllato da un sistema di autenticazione che verifica la reale identità degli utenti, distinguendoli tra amministratori e officine autorizzate.  Verranno utilizzati dei meccanismi di crittografia/sicurezza per l’invio di informazioni sensibili ( ad esempio password ). |

1.2.3 Criteri di Manutenzione

|  |  |
| --- | --- |
| Estendibilità | Il sistema deve essere progettato per accogliere al meglio nuove funzionalità, integrandole al meglio con i moduli già presenti. Quindi è necessario che il codice sia ben strutturato in moduli. |
| Modificabilità | Dovrà essere possibile effettuare modifiche al sistema in caso di bug.  In tal senso , come nel caso dell’estendibilità, la metodologia di programmazione aiuterà a risolvere vari problemi facilmente. |

1.2.4 Criteri per l’Utente Finale

|  |  |
| --- | --- |
| Usabilità | Il sistema dovrà essere User-Friendly, ossia di facile utilizzo e comprensione anche per l’utente neofita. La questione dell’usabilità risulterà meno importante per le officine, supponendo che gli addetti abbiano ricevuto una breve formazione sull’utilizzo del sistema ed utilizzandolo molte volte al giorno. |

**1.3 Definizioni, Acronimi e Abbreviazioni**

**JDBC:** Java DataBase Connection.

**DBMS:** Database Management System, Sistema di gestione del database

**Mysql:** DBMS libero e opensource.

**RAD:** Requirements Analysis Document.

**SDD:** System Design Documents

**User-friendly:** aggettivo utilizzato per definire un software di facile utilizzo anche per persone non esperte nell’utilizzo del computer

**Officina autorizzata:** Officina che ha richiesto di aderire al progetto CarCheck ed è stata accettata da un amministratore.

**1.4 Riferimenti**

* Documento RAD del progetto **CarCheck.**
* Object-Oriented Software Engineering Using UML, Patterns, and Java™ Third Edition Bernd Bruegge & Allen H. Dutoit.

**1.5 Panoramica**

Le attività di system design che costituiscono le fondamenta per l’architettura software del sistema:

1.Decomposizione del sistema: In questa prima fase il sistema viene scomposto in una serie di sottosistemi. Tale operazione è necessaria per ridurre la complessità in fase di design. Ciascuno di questi sottosistemi fornisce e riceve dei servizi da altri sottosistemi.

2. Mapping Hardware/Software: Vengono scelte le tecnologie e le piattaforme che il sistema andrà ad utilizzare. Vengono inoltre definiti una serie di elementi sempre relative ad aspetti architetturali.

3. *Gestione dei dati persistenti:* In questa fase vengono definiti i dati che dovranno essere memorizzati in modo permanente, le tecnologie e le modalità con

4.Politiche di accesso e sicurezza: In questa fase si definiscono i meccanismi di accesso in base alle varie tipologie di utente. Ciascun utente avrà un metodo di autenticazione (se necessario) e potrà svolgere una serie di operazioni a lui appropriate.

5. *Controllo del software globale:* che descrive il modo in cui è implementato il controllo globale del software e come si sincronizzano i sottosistemi.

6. *Condizioni Boundary:* che descriverà oltre l’avvio anche la gestione dei fallimenti, interruzione di corrente o anche a errori di progettazione.

**2.Current System Architecture**

Tra i sistemi precedenti a CarCheck figura in primo piano “Il portale dell’automobilista”. Tale piattaforma permette di eseguire una moltitudine di operazioni diverse tra loro, relative alla gestione dei veicoli, a pratiche ecc.

Il punto debole (secondo il nostro parere) è quello di individuare singole informazioni in parti diverse del sito. Infatti manca un meccanismo per reperire tramite una sola operazione dell’utente tutte le informazioni di un veicolo. L’utente è infatti costretto a navigare molte pagine per fare ciò.

**3. Architettura software proposta**

## 3.1 Panoramica

La scelta per l’architettura del nostro sistema è ricaduto su un sistema di tipo client/server. Un server gestirà tutto ciò che riguarda i dati persistenti e la loro accessibilità da parte degli utenti ,mentre il client si occuperà di visualizzare tali contenuti.

del client, e risponde in relazione al tipo di richiesta. In questa scelta siamo stati influenzati da criteri come performance e portabilità.

## 3.2 Decomposizione in sottosistemi

Per realizzare il sistema abbiamo scelto di seguire l’architettura Model-View-Controller (MVC) che verrà implementata utilizzando la tecnologia Servlet e JSP.

Essa si struttura in tre componenti, ognuno con un compito diverso all’interno del sistema.

* Il ***Model*** si occupa di gestire i dati persistenti. Contieni infatti metodi di lettura e scrittura su Database.
* La ***View*** si occupa di far visualizzare i dati all’utente e si occupa dell’interazione tra quest’ultimo ed il sistema.
* Il ***Controller*** riceve i comandi dell’utente attraverso le view e si occupa della logica di controllo dell’applicazione interagendo con gli altri due componenti.

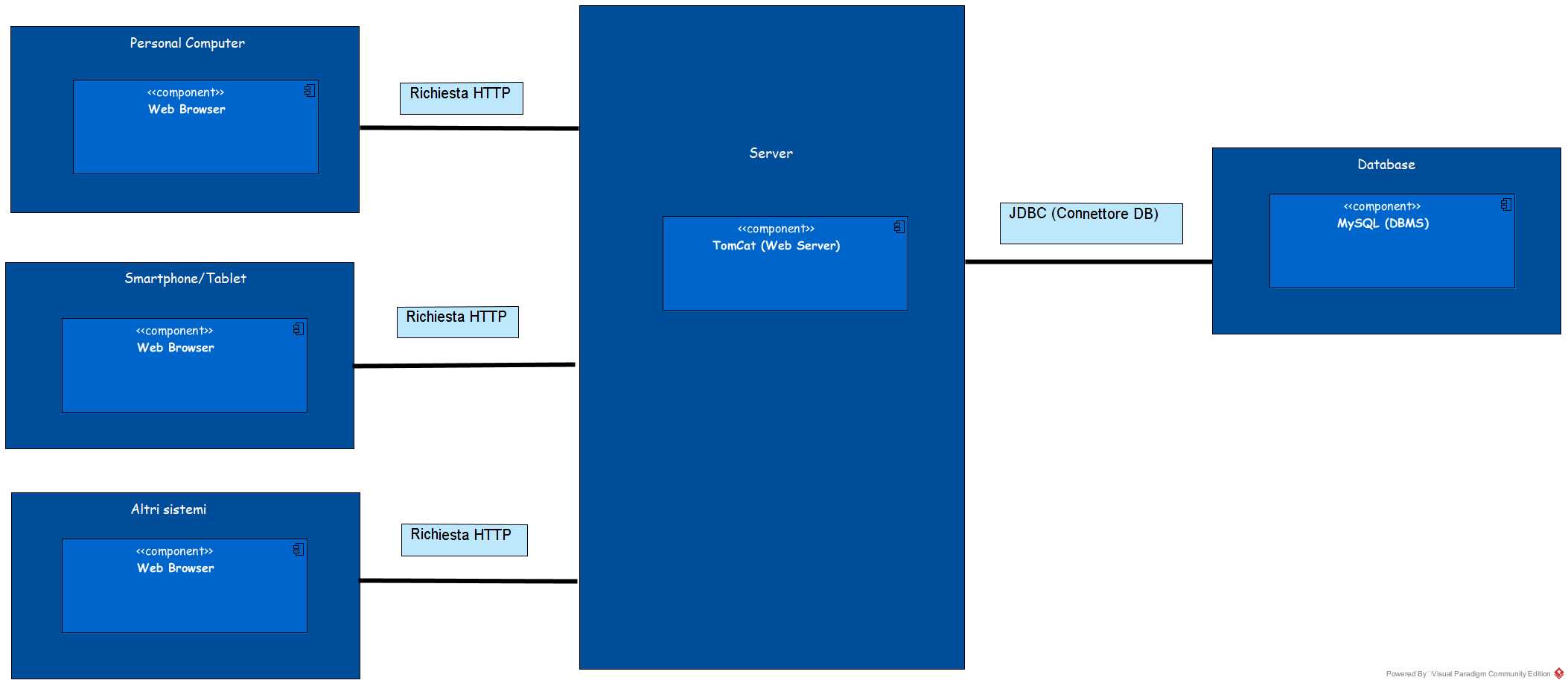
**3.3 Mapping Hardware/software**

Il Sistema Carcheck utilizzerà un’architettura Client/Server. Come server è stato scelto Apache Tomcat, un server locale in grado di simulare le varie operazioni di un reale server remoto.

Per quanto riguarda il DBMS, la scelta è ricadutoa su MySQL, essendo esso open-source e largamente diffuso.

I vari client saranno infine rappresentati da i vari dispositivi attraverso cui l’utente utilizza il sistema. Essi potranno essere un semplice pc, uno smartphone,un tablet, una smart tv o un qualsiasi altro dispositivo che abbia accesso ad internet mediante un web browser.

Lo schema in basso rappresenta ciò che è stato appena detto:



**3.4 Gestione dei dati persistenti**

**3.5 Controllo degli accessi e sicurezza**

Carcheck distingue i vari utenti in base alle operazioni che essi effettuano.

Il concetto alla base di tale suddivisione è che ciascuno di essi avrà obiettivi diversi ed utilizzerà il Sistema in un certo modo.

Un Utente non registrato vorrà semplicemente trovare informazioni rigurado ad un certo veicolo. Un’autofficina avrà bisogno di accedere al Sistema per gestire le revisioni e I lavori svolti. Infine l’amministratore avrà compiti relative alla gestione del Sistema. Ad esempio dovrà gestire le varie richieste delle Officine,accettarle/rifiutarle, nominare nuovi admin ecc.

Per controllare la reale identità di chi effettua l’accesso vengono utilizzate delle credenzali, ossia una email e una password. Se almeno una di esse risulterà errata nella fase di login, il Sistema dovrà impedire l’accesso a tale utente. In caso contrario, verrà create una nuova sessione che terminerà al momento del logout o dopo un tempo di inattività fissato.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Utente non registrato** | **Officina autorizzata** | **Amministratore** |
| **Utente** |  | * Login * Logout | * Login * Logout * Aggiungi nuovo utente * Rimuovi utente |
| **Veicolo** | * Ricerca dati veicolo | * Ricerca dati veicolo | * Ricerca dati veicolo * Modifica dati veicolo * Aggiungi veicolo * Rimuovi veicolo |
| **Revisione** |  | * Aggiungi revisione * Elimina revisione * Modifica dati revisione * Visualizza revisioni effettuate | * Elimina revisione * Modifica dati revisione * Visualizza revisioni effettuate da officina specifica |
|  |  |  |  |

**3.6 Global software control**

Il controllo del flusso del Sistema segue il meccanismo detto “Event-Driven”. Tale scelta è stata effettuata in quanto le funzionalità di Carcheck vengono scelte di volta in volta dall’utente ed impossibile prevederle. Tale strategia presenta numerosi benefici, essendo flessibile e facile da estendere.

**3.7 Boundary conditions**

Configurazione

Inizializzazione

Terminazione

Failure

**5.0 Glossario**

**Revisione:** Procedura di controllo di veicolo, finalizzata a verificarne le condizioni di sicurezza, il livello di emissioni inquinanti e la rumorosità, che devono risultare conformi agli standard prescritti da normative internazionali e nazionali.

**Client:** In una rete, ciascuno dei calcolatori che possono accedere ai servizi e alle risorse del server

**Server:** Elaboratore di elevate prestazioni che in una rete informatica fornisce un servizio ad altri elaboratori detti client.

**DBMS**: Sistema software progettato per la creazione, manipolazione e l’interrogazione efficiente di database.

**MySQL**: DBMS relazionale open-source

**Login\Logout :** Termini utilizzati per indicare la procedura di accesso e di disconessione ad un sistema informatico.

**Shutdown**: Termine con il quale si indica la fase di terminazione di un sistema informatico..

**Start-up:** Termina con il quale si indica la fase di avvio\accensione di un sistema informatico.