

SDD System Design Document

NewDM

|  |  |
| --- | --- |
| Riferimento |  |
| Versione | 0.8 |
| Data | 25/11/2020 |
| Destinatario | Prof.ssa F. Ferrucci |
| Presentato da | Cirillo Franco  Cirillo Luigi  Fusco Ciro  Aiello Vincenzo |
| Approvato da |  |

SDD System Design Document

**NewDM**

|  |  |
| --- | --- |
| Riferimento |  |
| Versione | 0.8 |
| Data | 25/11/2020 |
| Destinatario | Prof.ssa F. Ferrucci |
| Presentato da | Cirillo Franco  Cirillo Luigi  Fusco Ciro  Aiello Vincenzo |
| Approvato da |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Data | Versione | Cambiamenti | Autori |
| 28/11/2020 | Vr0.1 |  | Luigi Cirillo |

RevisionHistory

Sommario

1. [1. Introduzione 4](#_Toc57469040)

[1.1 Obiettivi del sistema 4](#_Toc57469041)

[1.2 Design Goals & Trade-offs 4](#_Toc57469042)

[1.3 Definizioni, acronimi e abbreviazioni 4](#_Toc57469043)

[1.4 Riferimenti 4](#_Toc57469044)

[1.6 Panoramica 4](#_Toc57469045)

1. [2. Architettura di Sistemi simili 4](#_Toc57469046)
2. [3. Architettura del Sistema proposto 4](#_Toc57469047)

[3.1 Panoramica 4](#_Toc57469048)

[3.2 Decomposizione in sottosistemi 4](#_Toc57469049)

[3.2.1 Diagramma di deployment 4](#_Toc57469050)

[3.3 Mapping hardware/software 4](#_Toc57469051)

[3.4 Gestione dati persistenti 4](#_Toc57469052)

[3.5 Controllo degli accessi e sicurezza 5](#_Toc57469053)

[3.6 Controllo flusso globale del sistema 5](#_Toc57469054)

[3.7 Condizione limite 5](#_Toc57469055)

[3.7.1 Start-up 5](#_Toc57469056)

[3.7.2 Terminazione 5](#_Toc57469057)

[3.7.3 Fallimento 5](#_Toc57469058)

1. [4. Servizi dei sottosistemi 5](#_Toc57469059)
2. [Glossario 5](#_Toc57469060)

# 1. Introduzione

## 1.1 Obiettivi del sistema

Lo scopo di ogni grande azienda che si rispetti è garantire ai propri clienti sicurezza e affidabilità. Per far questo ci affidiamo ad un sistema che ci permetta di gestire sempre in maniera smart tutte le condizioni per soddisfare i nostri clienti, e semplificare la vita dei nostri dipendenti. Nasce così NewDM!

Il sistema proposto parte dall’idea di poter fornire una piattaforma che sia in grado di gestire al meglio le varie interazioni tra i dipendenti e i clienti. Si vuole quindi realizzare una piattaforma desktop che consentirà lo svolgimento delle attività in maniera più agevole possibile per il bacino di utenza a disposizione.

NewDM avrà necessita di gestire dati persistenti: la scelta, per motivi di efficienza, ricade in un database relazionale che permetterà di tener traccia di tutte le informazioni necessarie di interesse dell’azienda. Da tale database attingerà un applicazione desktop deputata alla gestione delle interazioni con l’utente e alla manipolazione dei suddetti dati.

La piattaforma avrà un controllo per gli accessi alle varie aree(commesso, magazziniere, addetto assistenza) autenticando i propri utenti con uno username e password.

## 1.2 Design Goals & Trade-offs

Illustriamo nella seguente tabella gli obbiettivi di design per il sistema e le relative priorità (a numeri più bassi corrispondono priorità più elevate). Per ogni obiettivo riportiamo anche l’origine, facendo riferimento, in particolare, all’identificativo del requisito non funzionale ad esso associato.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Priorità** | **ID** | **Descrizione** | **Categoria** | **Origine** |
| 3 | DG\_1 | *Tempi di risposta*: Il sistema dovrà garantire che la scannerizzazione e la visualizzazione del prezzo del prodotto devono avvenire in al più sec. 2. Il sistema dovrà garantire che il calcolo totale e l’emissione dello scontrino devono avvenire in al più sec. 3. | Performance | RNF-P1  RNF-P2 |
| 1 | DG\_2 | *Robustezza*: Vogliamo proporre un sistema che abbia la capacità di sopravvivere ad input non validi. Pertanto, il sistema deve garantire il filtraggio dei dati inconsistenti o errati inseriti dall’utente, invitandolo a reinserirli. | Dependability | RNF-A1 |
| 3 | DG\_3 | *Disponibilità*: Il sistema deve essere sempre disponibile durante l’orario di lavoro (6am to 10pm). | Dependability | RNF-P4 |
| 1 | DG\_4 | *Tolleranza ai fault*: Il sistema dovrà evitare il verificarsi di più di un fault al giorno, in modo da garantire l’efficienza del servizio, ma soprattutto dovrà evitare ogni perdita di dati registrati. | Dependability | RNF-A2 |
| 1 | DG\_5 | *Sicurezza*: Il sistema dovrà fornire un metodo di autenticazione sicuro in modo che i dati siano protetti da accessi fraudolenti. | Dependability | RNF-A3 |
| 2 | DG\_6 | *Costi di sviluppo*: Lo sviluppo del prodotto richiederà costi ridotti sia in termini di risorse umane (per cui è fissato un tetto di 75 ore-lavoro), sia in termini economici (per cui si punta a ricorrere a soluzioni off-the-shelf open source) | Costo | Top management |
| 1 | DG\_7 | *Modificabilità*: Il sistema prodotto deve offrire una buona modificabilità, in modo da poter modificare le funzionalità esistenti. | Manutenibilità | RNF-S2 |
| 1 | DG\_8 | *Estensibilità*: Il sistema prodotto deve offrire una buona estensibilità, in modo da poter aggiungere nuove funzionalità al sistema | Manutenibilità | RNF-S3 |
| 2 | DG\_9 | *Leggibilità*: Il codice prodotto dev’essere semplice da comprendere. | Manutenibilità | RNF-S1 |
| 2 | DG\_10 | *Usabilità*: Il sistema deve essere facile da apprendere ed intuitivo da utilizzare senza necessariamente consultare la documentazione. | End User | RNF-U1  RNF-U2 |

*Tempo di rilascio vs Funzionalità*

Sebbene i tempi siano piuttosto proibitivi, preferiamo consegnare con leggero ritardo un prodotto che faccia ciò che promette piuttosto che un prodotto che non possa essere utilizzato a causa della mancanza di funzionalità.

*Prestazioni vs Costi*

Considerato il budget ridotto a disposizione, si preferisce rientrare nei costi dedicando un numero ridotto di ore-lavoro alla massimizzazione delle prestazioni.

*Prestazioni vs Affidabilità*

I dati gestiti dal sistema sono piuttosto sensibili, pertanto preferiamo garantire un maggior controllo di input e consistenza a scapito dei tempi di risposta.

## 1.3 Definizioni, acronimi e abbreviazioni

## 1.4 Riferimenti

* Requisiti funzionali: Sezione 3.2 del RAD
* Requisiti non funzionali: Sezione 3.3 del RAD
* Libro:

-- Object-Oriented Software Engineering (Using UML, Patterns, and Java) Third Edition

Autori:

-- Bernd Bruegge & Allen H. Dutoit

## 1.6 Panoramica

Nel documento verranno affrontati l’analisi delle architetture di sistemi simili, la decomposizione in sottosistemi del sistema proposto con la definizione della strategia di deploy e le condizioni limite. Verranno quindi definiti i servizi esposti da ciascun sottosistema.

# 2. Architettura di Sistemi simili

Nel nostro caso non è presente un’architettura software già esistente, pertanto abbiamo analizzato le architetture di sistemi simili.

Il primo software gestionale preso in considerazione è "EuroAzienda Supermercati". Esso è composto da due moduli software: Parte amministrativa, per la gestione del magazzino e per l’analisi e modifica prezzi. Punto cassa per la gestione dello scarico magazzino e per l’emissione dello scontrino.

Il secondo software analizzato è “Software Foodmanager”. E’ un software gestionale multipiattaforma in versione cloud html5 e in versione desktop. Permette la gestione dei permessi di ciascun operatore e prevede due moduli: il primo per la gestione del magazzino e il secondo per la gestione lato cassa.

Dall’analisi di questi due software si è evinto che alla base di tutte le piattaforme è presente un database per la memorizzazione e gestione dei prodotti presenti in magazzino. Quest’ultimo è interrogato da applicazioni desktop/web progettate con diverse tecnologie. Deduciamo ,quindi ,che ognuna delle suddette piattaforme sia stata sviluppata secondo un’architettura a 3 strati.

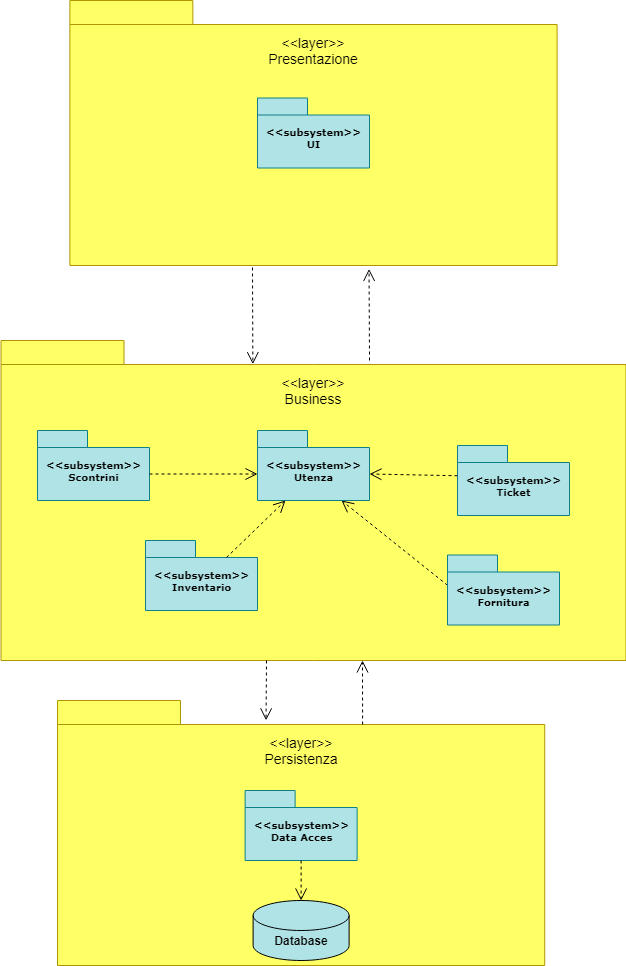
Nessuno dei software analizzati comprende la gestione dell’ area per l’assistenza clienti. Inoltre, soltanto il secondo analizzato, prevede un’autenticazione che permette di rendere disponibili, determinate funzionalità, solamente ad utenti autorizzati.

# 3. Architettura del Sistema proposto

## 3.1 Panoramica

NewDM è Piattaforma desktop che fornisce un’interfaccia per le sue funzionalità. Essa, inoltre, ricorre all’utilizzo di un database relazionale per il salvataggio dei dati persistenti.

## 3.2 Decomposizione in sottosistemi



Il sistema è suddiviso in 3 livelli logici: presentazione, business e persistenza che si occupano rispettivamente di presentazione delle informazioni all’utente, definizione della logica applicativa e gestione dei dati persistenti.

Il livello di presentazione è composto da un solo sottosistema:

* UI: definisce l’interfaccia utente

Il livello di business è composto da cinque sottosistemi

* Utenza: definisce l’utente (cassiere, magazziniere, addetto assistenza) ed offre i relativi servizi di autenticazione.
* Scontrini: modella tutto ciò che riguarda il processo di acquisto dei prodotti ed emissione scontrino
* Ticket: modella il processo di gestione ticket e tutte le sue operazioni
* Inventario: modella i prodotti nel magazzino e le relative operazioni
* Fornitura: modella il processo di richiesta prodotto
* Data Access: si occupa del reperimento e salvataggio delle informazioni manipolate dal sistema.

La suddivisione in sottosistemi è stata quindi realizzata tramite una strutturazione 3-tier, in modo specifico, sulla base del pattern MVC. La gestione di View e Controller è affidata al livello di presentazione, mentre la gestione del Model è affidata al livello di business e al livello di persistenza.

### 3.2.1 Diagramma di deployment

NewDM si compone di due elementi essenziali: un’applicazione desktop e un database MySQL eseguito su un server.

L’architettura scelta prevede l’interazione dell’applicazione desktop con il database: queste due componenti sono installate nella stessa rete locale e si scambiano informazioni tramite un driver di protocollo di rete, ossia, JDBC.

Immagine che contiene tavolo

Descrizione generata automaticamente

## 3.3 Mapping hardware/software

NewDM si compone di due componenti principali:

* Piattaforma desktop, cui saranno allocati i layer di presentazione e business oltre al sottosistema di data access
* Database, realizzante il layer di persistenza

Il sistema necessita di diverse macchine con sistema operativo Windows al fine di garantire l’operabilità della suddetta piattaforma desktop.

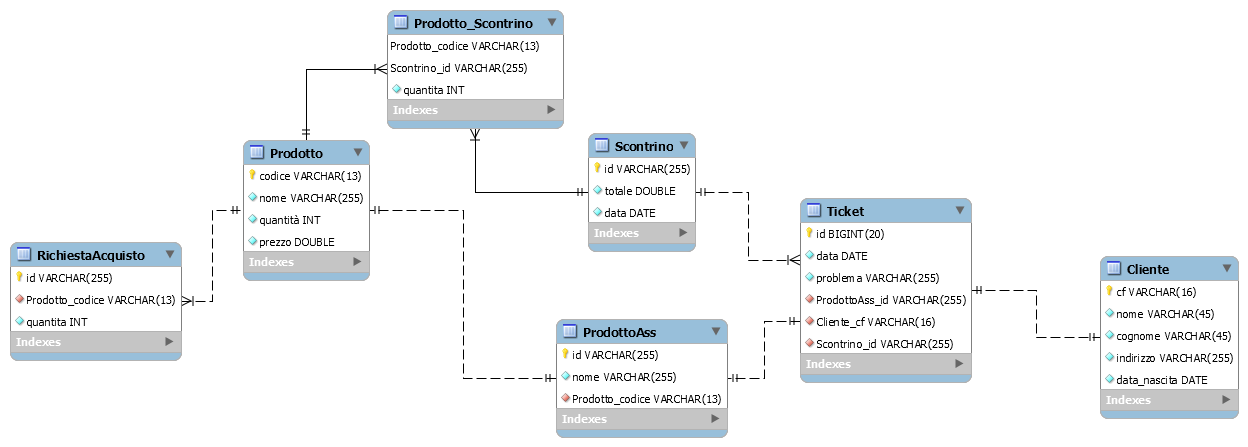
Il sistema inoltre necessita di una macchina in grado di supportare MySQL per garantire l’operabilità del database con cui le diverse piattaforme desktop si interfacciano.

La comunicazione avverrà tramite protocollo jdbc.

Per la connessione alla rete locale verranno utilizzati cavi ethernet.

## 3.4 Gestione dati persistenti

Per la gestione dei dati persistenti, NewDM si affida ad un database relazionale gestito tramite MySQL. La struttura dei dati memorizzati segue il seguente schema:



## 3.5 Controllo degli accessi e sicurezza

Il controllo di accesso è garantito tramite l’utilizzo di username e password per gli utenti che hanno accesso a contenuti sensibili, per prevenire eventuali creazioni o modifiche a oggetti che modellano entità del dominio, in modo da prevenire accessi non autorizzati a dati sensibili. Inoltre, almeno nella prima versione non saranno implementati metodi di recupero o modifica password.

Per scelte progettuali, il salvataggio delle password sarà in chiaro sul database: non ci sarà nessun tipo di cifratura, almeno nella versione iniziale del sistema.

Le operazioni che gli utenti della piattaforma desktop possono effettuare sugli oggetti sono riportati nella tabella che segue:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Oggetti  Attore | Utenza | Prodotto | Scontrino | Ticket | Richiesta  fornitura |
| Cassiere | Autenticazione  Logout | Scanner prodotto | Stampa scontrino |  |  |
| Magazziniere | Autenticazione  Logout | Richiesta nuovo prodotto |  |  | Richiesta  fornitura |
| Addetto Assistenza | Autenticazione  Logout |  |  | Crea nuovo ticket per prodotti in garanzia |  |

Dalla tabella si evince come l’utente non abbia interazione dirette con il controllo di accesso al database, cui accedono i singoli sottosistemi di business: per questo motivo si è deciso di non riportare nella matrice d’accesso.

## 3.6 Controllo flusso globale del sistema

Il sistema adotta un controllo del flusso globale del tipo Procedure-Driven, permettendo a tutti gli attori di interagire con un controllore il quale per garantire l’integrità dei dati e la persistenza prende in carico tutte le azioni e, va a modificare il database.

## 3.7 Condizione limite

### 3.7.1 Start-up

### 3.7.2 Terminazione

### 3.7.3 Fallimento

# 4. Servizi dei sottosistemi

# Glossario

*DBMS*: sistema software per la gestione dei dati persistenti su database

*MySQL*: specifico DBMS