System Design Document

Progetto:

CleanDesk

CleanDesk
Powered by AI

Riferimento:	
Versione:	0.6
Data:	15/12/2023
Destinatario:	Esame di Ingegneria del Software 2023/24
Presentato da:	Ambrosio Gennaro, Camoia Andrea
Approvato da:	

UNIVERSITÀ DEGLI STUDI
DI SALERNO

Revision Hystory

Data	Versione	Descrizione	Autori
04/12/2023	0.1	Stesura iniziale del documento, aggiunta introduzione e definizione Architettura Corrente	Ambrosio Gennaro Camoia Andrea
05/12/2023	0.1	Aggiunta Design Goal e relativi Trade-off	Ambrosio Gennaro Camoia Andrea
05/12/2023	0.2	Definizione dei Sottosistemi	Ambrosio Gennaro Camoia Andrea
06/12/2023	0.2	Aggiunta di Component Diagram e Diagramma Architetturale	Ambrosio Gennaro Camoia Andrea
07/12/2023	0.2	Aggiunta del Mapping Hardware-Software	Ambrosio Gennaro Camoia Andrea
08/12/2023	0.3	Definizione Controllo degli Accessi e Controllo Globale del Sistema	Ambrosio Gennaro Camoia Andrea
08/12/2023	0.4	Descrizione dei Servizi dei Sottosistemi	Ambrosio Gennaro Camoia Andrea
10/12/2023	0.5	Definizione della Gestione dei dati persistenti.	Ambrosio Gennaro Camoia Andrea
11/12/2023	0.6	Definizione delle Condizioni Limite del sistema.	Ambrosio Gennaro Camoia Andrea
11/12/2023	0.6	Definizione del Glossario.	Ambrosio Gennaro Camoia Andrea
15/12/2023	0.6	Modifica Boundary Case	Ambrosio Gennaro Camoia Andrea

Team Member

Nome	Ruolo nel progetto	Acronimo	Informazioni di contatto
Andrea Camoia	Team Member	AC	a.camoia@studenti.unisa.it
Gennaro Ambrosio	Team Member	AG	g.ambrosio35@studenti.unisa.it

Indice

1 Introduzione

1.1 Scopo del Sistema

Il software CleanDesk è stato progettato per migliorare l'esperienza informatica e massimizzare la produttività dei propri utenti.

A questo scopo saranno forniti strumenti avanzati, che sfruttano l'intelligenza artificiale per organizzare e trovare rapidamente i documenti, immagini e progetti all'interno di un ambiente digitale come il Desktop.

CleanDesk vuole semplificare la vita digitale di qualsiasi suo utente: dal professionista, all'appassionato di tecnologia fino ad arrivare agli utilizzatori casuali e poco pratici in materia; a tutti sarà fornito un ambiente digitale privo di caos ed ordinato in maniera intuitiva. Inoltre ognuno potrà gestire questo spazio in maniera consapevole tramite un efficiente gestione ed analisi della memoria fornita da CleanDesk.

1.2 Obiettivi di Design (Design Goals)

In questo paragrafo vengono esposti i Design Goals ("Obiettivi di Design"), che descrivono le qualità chiave del sistema (quelle che devono essere ottimizzate) e stabiliscono le loro priorità.

Sviluppando in modo corretto i Design Goal sarà possibile stabilire un percorso di sviluppo ben delineato, solido e dettagliato.

Nella tabella che segue, i design goal sono descritti in base ai seguenti campi, ognuno corrispondente a una colonna:

- **Rank**: Specifica un valore di priorità che va da 1, con massima priorità, ad *N*, con priorità minima, dove *N* è il numero di Design Goals individuati.
- **ID**: Acronimo utile ad identificare in modo specifico un Design Goal;
- **Descrizione**: utile per la comprensione del Design Goal;
- **Categoria**: raggruppa i design goal che descrivono qualità che rientrano nella stessa macro-area;
- **RNF di Origine**: Requisito NON Funzionale da cui è scaturita la creazione del Design Goal;

In particolare, i design goal sono stati divisi nelle categorie descritte successivamente:

- **Performance**: : indica, in maniera quantificabile, il livello di prestazioni del sistema software;

- **Dependability**: la categoria relativa all'affidabilità del sistema.
- **Maintenance**: indica, quanto effort è necessario per apportare modifiche al sistema dopo il primo deploy;
- **End User**: Includono qualità che sono desiderabili dal punto di vista dell'utente, ma che non sono state coperte dai criteri di performance e dependability;

Tabella dei Design Goal

Rank	ID	Descrizione	Categoria	RNF di Origine
1	DG_01 Integrità	Il sistema dovrà essere in grado di rilevare file riservati al Sistema Operativo, evitando di effettuare l'organizzazione su di essi, al fine di non comprometterne l'integrità.	Dependability	RNF_RLB_01
2	DG_02 Riservatezza	Il sistema lavorerà sempre in locale, non salvando alcuna informazione dei file analizzati né utilizzando queste per scopi di terze parti	Dependability	RNF_LGL_02
3	DG_03 Tempi di Risposta	Il sistema dovrà impiegare massimo 5 secondi per analizzare un file.	Performance	RNF_PRF_02
4	DG_04 Interfaccia minimale	Il sistema dovrà garantire un'interfaccia semplice ed intuitiva affinché l'utente possa essere in grado di poter usufruire di tutte le funzionalità che il sistema offre	End User	RNF_USG_02
5	DG_05 Manutenibilità del Sistema	Il sistema deve essere grado di garantire una facile manutenibilità ed estensione grazie alla modularità e alla leggibilità del suo codice	Maintenance	RNF_SPT_03

Trade-off

Al fine di soddisfare i Design Goals appena specificati, saranno applicati i seguenti Trade-off:

Trade-off	Razionale
Velocità vs Integrità.	Il sistema verrà ottimizzato per garantire una maggiore precisione nel controllo dell'integrità dei file del Sistema Operativo e di software esterni, a discapito di una leggera riduzione nella velocità di esecuzione.
Tempo di rilascio vs funzionalità.	Se i tempi di rilascio sono stringenti, possono essere rilasciate meno funzionalità di quelle richieste, ma nei tempi giusti.
Tempo di rilascio vs. qualità.	Se i tempi di rilascio sono stretti, la consegna del prodotto software potrà subire dei ritardi, prediligendo la qualità e correggendo tutti i Bug individuati.

1.3 Definizioni, acronimi e abbreviazioni

Nella seguente sezioni si presentano e si riportano le definizioni adoperate nel seguente documento:

- **Sottosistema**: un sottoinsieme dei servizi del dominio applicativo, formato da servizi legati da una relazione funzionale.
- **Design Goal**: le qualità sulle quali il sistema deve essere focalizzato.
- **Dati Persistenti**: dati che sopravvivono all'esecuzione del programma che li ha creati e che dunque vengono salvati.
- **Mapping Hardware/Software**: studio della connessione tra parti fisiche e logiche di cui si compongono il sistema.
- **SDD**: System Design Document.
- **RAD**: Requirements Analysis Document.
- **DG**: Design Goal;

1.4 Riferimenti

1. Dispense del prof. Carmine Gravino, fornite mediante la pagina del corso "Ingegneria del Software - Resto 2 - 2023/2024" sulla Piattaforma E-learning del Corso di Laurea in Informatica dell'Università degli Studi di Salerno;

2. Libro di testo "Object Oriented Software Engineering Using UML Patterns and Java Prentice Hall 2010 Bernd Bruegge Allen H.Dutoit"

3. Libro di testo "C. GHEZZI, D. MANDRIOLI, M. JAZAYERI, INGEGNERIA DEL SOFTWARE – FONDAMENTI E PRINCIPI, PRENTICE HALL, 2004"

4. CleanDesk_RAD

5. CleanDesk_SoW

1.5 Organizzazione del documento

Il presente documento di System Design consta di quattro sezioni:

1. **Introduzione**: Viene descritto in generale lo scopo del sistema, gli obiettivi di design che il sistema propone di raggiungere ed eventuali trade-off, cioè compromessi per il raggiungimento di quest'ultimi.

2. **Architettura software corrente**: Viene descritto lo stato attuale dell'architettura del software già presente.

3. **Architettura software proposta**: Viene descritto come il sistema sarà definito e partizionato in sottosistemi, il loro mapping Hardware/Software, la gestione dei dati persistenti. Verranno poi presentate la struttura dei singoli sottosistemi e le boundary conditions riguardanti l'intero sistema.

4. **Servizi dei Sottosistemi**: Descrive i servizi forniti da ciascun sottosistema in termini di operazioni. Essa funge da riferimento per i team utile a stabilire confini tra i sottosistemi.

5. **Glossario**: Contiene la lista dei termini usati nel documento con annessa spiegazione.

2 Architettura del Sistema Corrente

Attualmente sul mercato esistono due piattaforme che assolvono, in parte, alle funzionalità che CleanDesk intende fornire:

- Abelssoft File Organizer [1]: un software che permette di organizzare i file all'interno di una cartella, seguendo però regole di basso livello e poco "intelligenti".
- FileFusion by Abelssoft[2]: che permette di migliorare l'utilizzo della memoria offrendo l'unica funzionalità di individuazione dei file duplicati.

CleanDesk invece intende includere, e allo stesso tempo migliorare, le funzionalità proposte da questi due software all'interno di un unico sistema. Il quale sarà in grado di fornire un Organizzazione dei File ed una Gestione della Memoria efficiente, intuitiva e più flessibile per l'utente.

[1] https://www.abelssoft.de/en/windows/helpers/abelssoft-file-organizer
[2] https://www.abelssoft.de/en/windows/system-utilities/filefusion

3 Architettura del Sistema Proposto

3.1 Panoramica

CleanDesk si basa su un **architettura software a "Three-Tier"**, con una stratificazione dei sottosistemi in appunto 3 livelli indipendenti:

- **Interface Layer**, include tutti i boundary object che interfacciano con l'utente.
- **Application Logic Layer**, include tutti gli oggetti relativi al controllo e alle entità che realizzazione l'elaborazione, le regole di verifica e notifica richiesta dall'applicazione.
- **Storage Layer**, effettua la memorizzazione, il recupero e l'interrogazione di oggetti persistenti.

Questo modello architetturale è stato preferito in quanto la completa indipendenza tra i livelli, ha il vantaggio di fornire una separazione logica delle componenti software così da aumentarne la manutenibilità, la scalabilità ed il riutilizzo del codice.

3.2 Decomposizione in Sottosistemi

Al fine di scomposizione, stati individuati i seguenti sottosistemi:

- **Organizzazione File:** sottosistema che si occupa delle funzionalità relative all'avvio di una nuova organizzazione dei file e la visualizzazione di un report per un organizzazione precedente.
- **Analisi Memoria:** che si occupa delle funzionalità di monitoraggio e analisi dello spazio di memoria occupato dai file in una cartella.
- **Ricerca Duplicati:** offre le funzionalità per individuare file simili e/o duplicati all'interno di una cartella.
- **Gestione File System:** sottosistema che si occupa creare, leggere, analizzare e spostare File o Cartelle del File System del Computer sul quale il software esegue.
- **Storage:** sottosistema responsabile della gestione dei dati persistenti.
- **StorageConnection:** che si interpone tra quello di Storage e gli altri sottosistemi.

Component Diagram

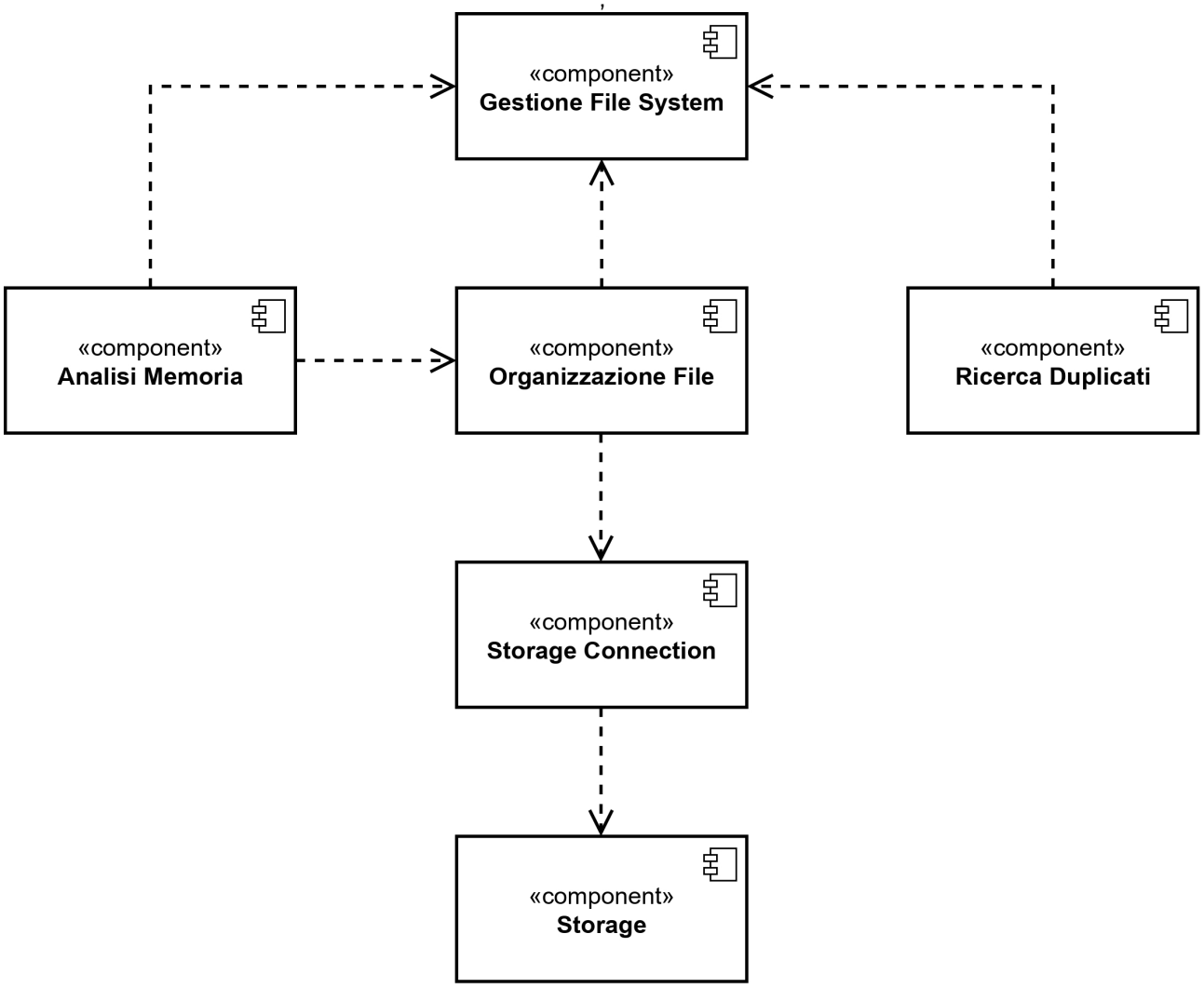


Figure 3.1: **UML Component Diagram**

Diagramma Architetturale

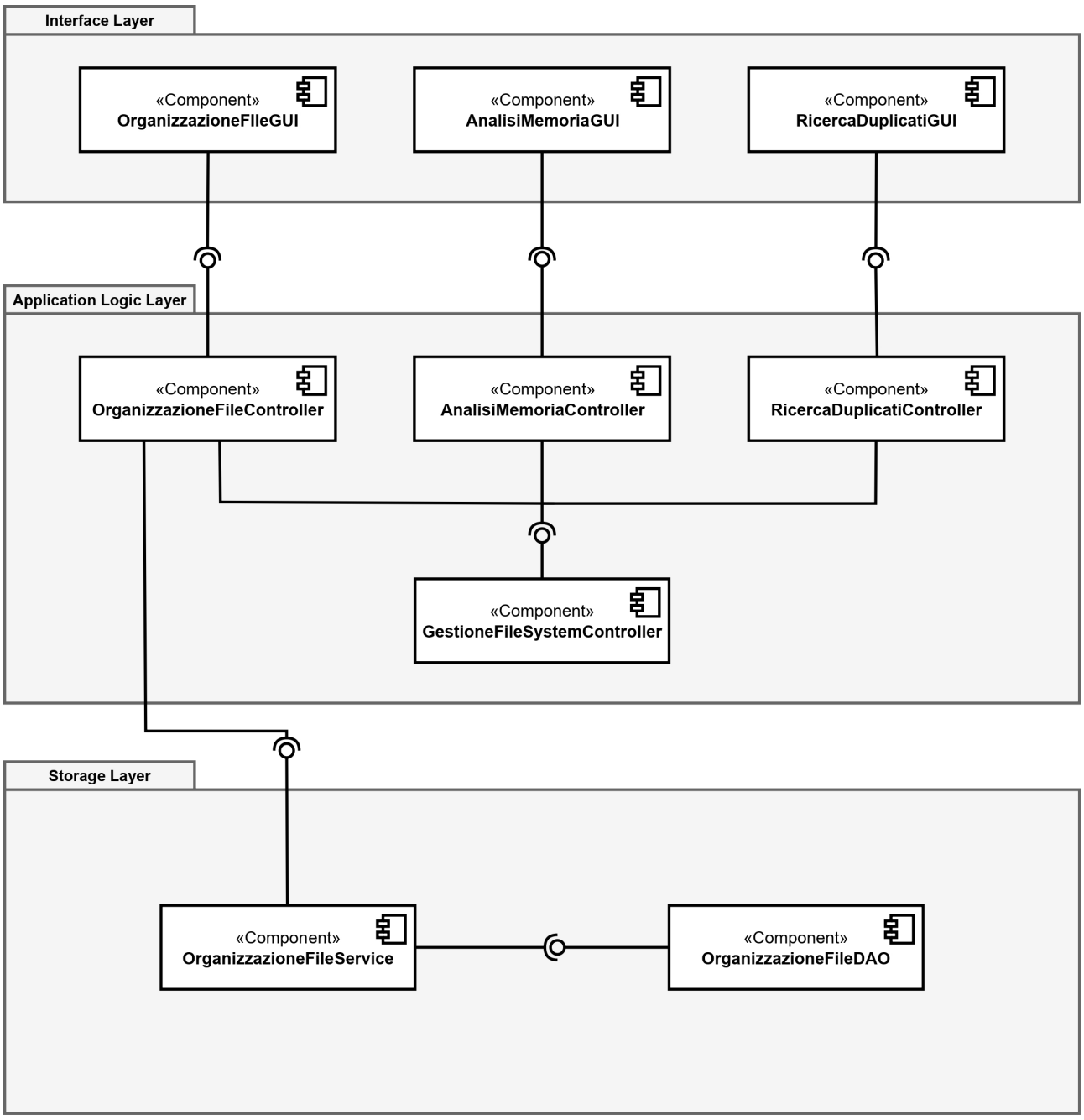


Figure 3.2: **Diagramma Architetturale**

3.3 Mapping Hardware/Software

CleanDesk sarà sviluppato come un Software scaricabile ed installabile su qualsiasi Computer con sistema operativo Windows o MacOS.
Il programma non richiede una connessione ad internet per il suo funzionamento, la quale è necessaria solo per scaricare il file di installazione dall'apposito portale o altre fonti.

Infine, i dati persistenti saranno gestiti tramite un database locale, incluso nell'applicazione.

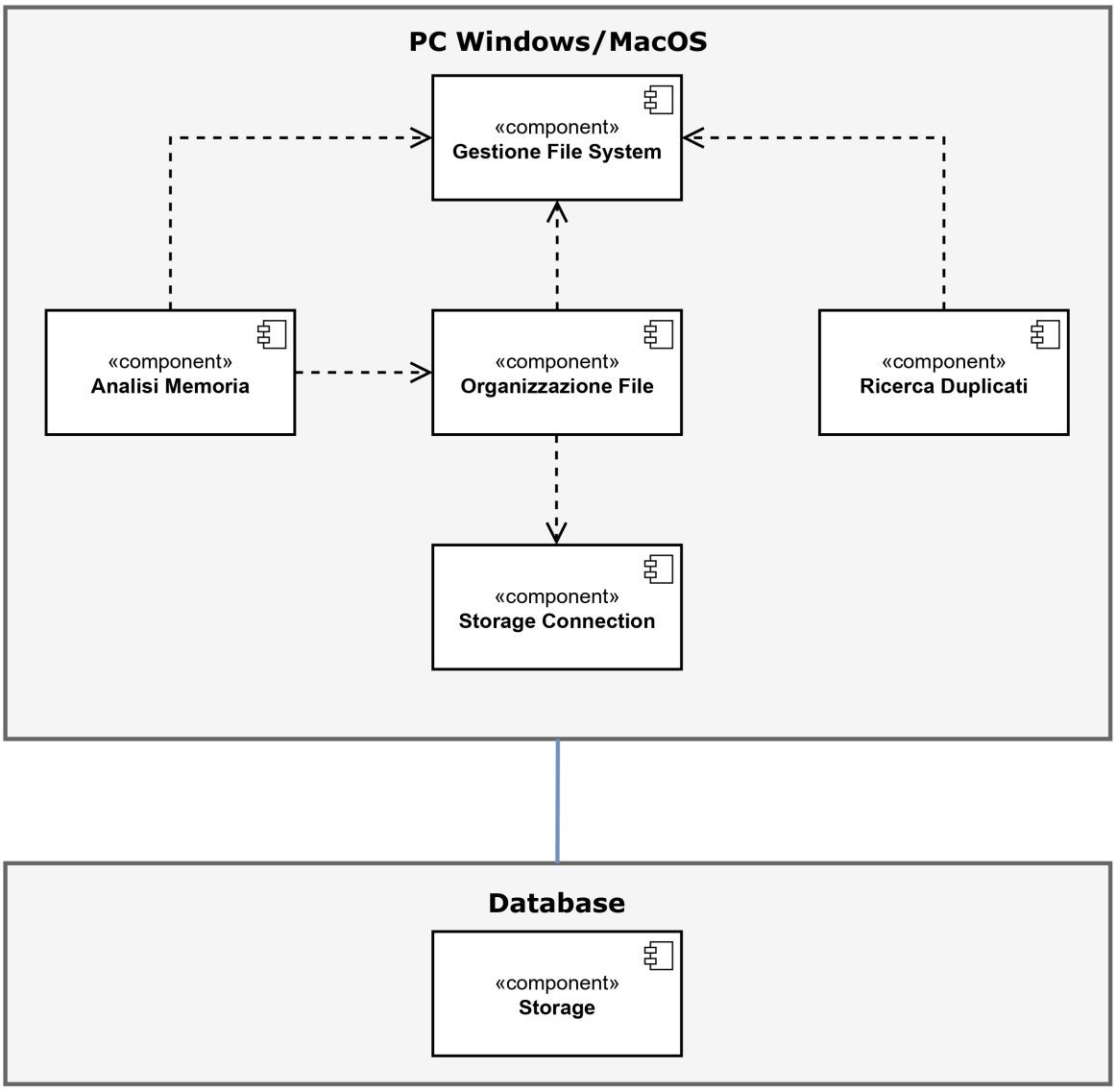


Figure 3.3: **UML Deployment Diagram**

3.4 Gestione dei dati persistenti

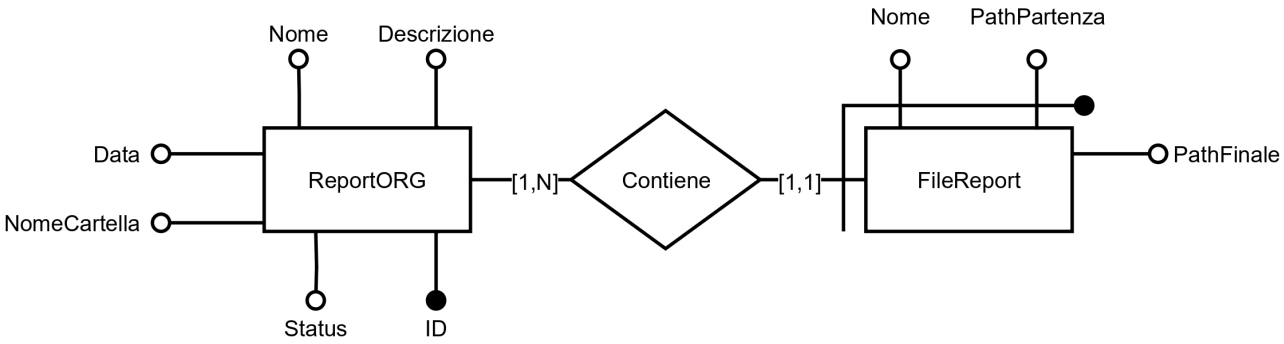
Per la gestione dei dati persistenti del sistema, è stata scelta l'implementazione di un database relazionale, in particolare **SQLite3**[3], che è un sistema di gestione di database relazionale (RDBMS) leggero e incorporato. Questa è una libreria software scritta in linguaggio C che fornisce un database SQL autonomo senza richiedere un processo server separato e senza dipendenze esterne.
È stato scelto questo sistema, per i seguenti motivi:

- **Autonomo**: A differenza di molti altri sistemi di gestione di database, SQLite3 non richiede un processo server separato o un'installazione complessa. I dati vengono archiviati in un singolo file, rendendo SQLite3 facile da distribuire e integrare nelle applicazioni.
- **Zero configurazione**: Non è necessario configurare o gestire un server separato e l'accesso ai dati avviene direttamente attraverso chiamate di funzioni API.
- **Leggero**: SQLite3 è progettato per essere leggero in termini di risorse di sistema richieste, ciò lo rende adatto per applicazioni mobili, embedded e in ambienti con risorse limitate.
- **Transazionale**: Supporta transazioni ACID (Atomicity, Consistency, Isolation, Durability), garantendo la coerenza e l'integrità dei dati durante le operazioni di inserimento, aggiornamento e cancellazione.
- **Velocità**: SQLite3 è noto per le sue prestazioni efficienti, specialmente in scenari di lettura intensiva. Le prestazioni possono essere notevoli anche in scenari di scrittura moderata.
- **Open Source**: SQLite3 è un software open source, distribuito sotto la licenza di dominio pubblico, il che significa che può essere utilizzato liberamente senza dover pagare royalty o licenze.

Date queste caratteristiche, SQLite3 si candida come la scelta migliore per un software desktop come CleanDesk.

[3] `https://docs.python.org/3/library/sqlite3.html`

Schema Entità Relazione



- ReportORG(**ID**, Nome, Descrizione, NomeCartella, Data, Status);
- FileReport(**Report**, **Nome**, **PathPartenza**, PathFinale);

Dizionario dei dati

Di seguito, verranno mostrati gli attributi e le entità individuate nello schema Entità-Relazione.

Nome Entità	**ReportORG**		
Descrizione	Report finale di un organizzazione dei file effettuata tramite il sistema.		
Nome campo	**Tipo**	**Vincoli di chiave**	**Altri vincoli**
ID	INT	PRIMARY KEY	NOT NULL, AUTOINCREMENT
Nome	VARCHAR(30)		NOT NULL, DEFAULT ("Organizzazione_{ID}")
Descrizione	VARCHAR(144)		DEFAULT
NomeCartella	VARCHAR(30)		NOT NULL
Data	DATE		NOT NULL
Status	VARCHAR(10)		NOT NULL

Table 1: Dizionario dei Dati per l'entità **RerportORG**

Nome Entità		**FileReport**	
Descrizione		Report relativo all'organizzazione di un singolo file.	
Nome campo	**Tipo**	**Vincoli di chiave**	**Altri vincoli**
Report	INT	FOREIGN KEY, PRIMARY KEY	NOT NULL
Nome	VARCHAR(64)	PRIMARY KEY	NOT NULL
PathPartenza	VARCHAR(144)	PRIMARY KEY	NOT NULL
PathFinale	VARCHAR(144)		NOT NULL

Table 2: Dizionario dei Dati per l'entità **FileReport**

3.5 Controllo degli accessi e sicurezza

Nella seguente sezione si specificano i permessi per accedere ai servizi offerti dal sistema.

Oggetto \ Attore	Utente	Sistema
Organizzazione File	AvviaNuovaOrganizzazione VisualizzaStoricoOrganizzazioni VisualizzaDettagliReport	-
Analisi Memoria	AvviaAnalisiMemoria	-
Ricerca Duplicati	AvviaRicercaFileDuplicati	-
Gestione File System	-	SpostaFile CreaCartella LeggiFile

3.6 Controllo globale del software

L'interazione con il sistema CleanDesk avviene tramite un interfaccia grafica, mediante la quale l'utente può avviare una determinata funzionalità. A seconda del comando impartito, il sistema avvia il controllo appropriato per l'esecuzione e gestione delle funzionalità corrispondenti.

Di conseguenza, CleanDesk ha un controllo di flusso di tipo **Event-Driven**, in cui appunto le esecuzioni dipendono da un evento generato.

3.7 Condizioni limite

In questa sezione viene definito il comportamento del sistema nelle condizioni limite di: Avvio, Spegnimento, Fallimento del sistema ed Errore di accesso ai dati persistenti.

Avvio del Sistema

Identificativo *UC_BC_01*	Avvio del sistema.	**Data**	10/12/23
		Versione	0.8
		Autore	Andrea Camoia

Descrizione	Lo *UC* descrive il comportamento del sistema durante l'avvio
Attore Principale	**Sistema**
Attori secondari	**Utente**
Entry Condition	Il sistema è installato e pronto per essere avviato. **AND** L'utente è intenzionato ad avviare il software.
Exit condition *on success*	Il sistema viene avviato correttamente
Exit condition *on failure*	Il sistema non si avvia.

FLUSSO DI EVENTI PRINCIPALE/MAIN SCENARIO		
1	**Utente:**	Richiede di avviare il software cliccando sulla relativa icona.
2	**Sistema:**	Si avvia andando a visualizzare l'interfaccia grafica e rendendo disponibili le funzionalità e tutti i servizi.
I Scenario/Flusso di eventi di ERRORE: Il sistema non riesce ad avviarsi correttamente		
2.1	Sistema	Visualizza un messaggio di errore all'utente. Il messaggio segnala che non è stato possibile avviare correttamente il software.

Spegnimento del Sistema

Identificativo *UC_BC_02*	Spegnimento del sistema	**Data**	10/12/23
		Versione	0.8
		Autore	Andrea Camoia

Descrizione	Lo *UC* descrive il comportamento del sistema durante l'avvio
Attore Principale	**Sistema**
Attori secondari	**Utente**
Entry Condition	Il Sistema è stato avviato correttamente **AND** Il Sistema non sta eseguendo nessuna funzionalità **AND** Il Sistema non è ancora stato spento
Exit condition *on success*	Il sistema viene arrestato correttamente
Exit condition *on failure*	Il sistema non viene arrestato correttamente.

FLUSSO DI EVENTI PRINCIPALE/MAIN SCENARIO		
1	**Utente:**	Esegue il comando per spegnere il sistema.
2	**Sistema:**	Avvia lo spegnimento rilasciando tutte le risorse utilizzate.

Errore di Accesso ai Dati Persistenti

Identificativo *UC_BC_03*	Errore di Accesso ai dati persistenti.	**Data**	10/12/23
		Versione	0.8
		Autore	Andrea Camoia

Descrizione	Lo *UC* descrive il comportamento del sistema nel caso in cui non riesca ad accedere ai dati persistenti o questi risultano essere corrotti
Attore Principale	**Sistema**
Attori secondari	**Utente**
Entry Condition	Il sistema non riesce ad accedere ai dati persistenti **OR** I Dati persistenti sono corrotti.
Exit condition *on success*	Il sistema continua la sua normale esecuzione.
Exit condition *on failure*	Il sistema non riesce a funzionare correttamente.

FLUSSO DI EVENTI PRINCIPALE/MAIN SCENARIO		
1	**Sistema:**	Termina l'esecuzione della funzionalità corrente.
2	**Sistema:**	Comunica all'utente l'impossibilità di accedere ai dati persistenti.
3	**Sistema:**	Invita l'utente a riavviare il sistema riprovando più tardi.

Organizzazione Interrotta

Identificativo *UC_BC_04*	Organizzazione Interrotta	**Data**	11/12/23
		Versione	0.8
		Autore	Andrea Camoia

Descrizione	Lo *UC* descrive il comportamento del sistema nel caso in cui la funzionalità di organizzazione dei File viene interrotta durante l'esecuzione.
Attore Principale	**Sistema**
Attori secondari	NA
Entry Condition	A causa di fattori esterni, la funzionalità di organizzazione viene interrotta durante l'esecuzione.
Exit condition *on success*	Il sistema ritorna in uno stato funzionante.
Exit condition *on failure*	Il sistema non ritorna in uno stato funzionante

FLUSSO DI EVENTI PRINCIPALE/MAIN SCENARIO		
1	**Sistema:**	Include US_BC_01 - Avvio del sistema
2	**Sistema:**	Mostra un messaggio di errore, comunicando all'utente che la funzionalità di Organizzazione è stata interrotta improvvisamente.
3	**Sistema:**	Ripristina la posizione dei File a quella originaria (prima dell'organizzazione).
4	**Sistema:**	Visualizza un report di Errore dell'organizzazione interrotta.
I Scenario/Flusso di eventi di ERRORE: Il sistema non riesce a recuperare le informazioni dell'organizzazione interrotta.		
2.1	Sistema	Visualizza un messaggio di errore generico all'utente.

4 Servizi dei Sottosistemi

In questa sezione vengono descritti i servizi di ogni sottosistema precedentemente elencati.

Sottosistema Organizzazione File

Servizio	Descrizione	Interfaccia
Avvia nuova organizzazione	Questa funzionalità permette all'utente di avviare una nuova organizzazione dei file in una cartella specificata.	OrganizzazioneFileService
Visualizza Storico Organizzazioni	Questa funzionalità permette all'utente di visualizzare lo storico dei report di tutte le Organizzazioni fatte in precedenza	OrganizzazioneFileService
Visualizza Dettagli Report Organizzazione	Questa funzionalità permette all'utente di visualizzare i dettagli del Report di un Organizzazione Precedente.	OrganizzazioneFileService

Sottosistema Analisi Memoria

Servizio	Descrizione	Interfaccia
Avvia analisi della memoria	Questa funzionalità permette all'utente di effettuare un analisi sull'utilizzo della memoria di una specifica cartella, in base alle categorie di file presenti.	AnalisiMemoriaService

Sottosistema Ricerca Duplicati

Servizio	Descrizione	Interfaccia
Avvia ricerca file duplicati/simili	Questa funzionalità permette all'utente di avviare una ricerca di file simili/duplicati presenti all'interno di una specifica cartella	RicercaDuplicatiService

Sottosistema Gestione File System

Servizio	Descrizione	Interfaccia
Sposta File	Funzionalità che permette di modificare la posizione (path) di un file.	GestioneFileSystemService
Leggi Contenuto File	Funzione che permette di leggere il contenuto di un file	GestioneFileSystemService
Crea Cartella	Funzione che permette di creare una nuova cartella assegnandogli un nome.	GestioneFileSystemService

5 Glossario

Sigla/Nome	Descrizione
Trade-off	È una scelta che comporta un compromesso, dove l'ottenimento di un beneficio o di un vantaggio è associato alla perdita o al sacrificio di un altro aspetto.
File System	È una struttura organizzativa utilizzata dai computer per memorizzare, organizzare e recuperare file.
Event-Driven	Meccanismo di gestione del controllo globale in cui il flusso del programma è determinato e gestito tramite eventi
Sistema embedded	È un sistema informatico dedicato a una specifica funzione o applicazione all'interno di un sistema più ampio.
ACID	Acronimo che rappresenta un insieme di proprietà fondamentali garantite da un sistema di gestione di database relazionale.
Open Source	È un modello di sviluppo di un software in cui il codice sorgente del programma è reso pubblicamente disponibile ed accessibile a tutti.
Royalty	Si riferisce ad un pagamento periodico che viene erogato come compenso per l'utilizzo di un determinato bene.