

**Documentazione Progetto di**

**Progettazione, Algoritmi e Computabilità**

Pellegrinelli Nico 1065869

Pellegrinelli Sean 1065868

Anno Academico 2022/2023

Sommario

[Iterazione 0 3](#_Toc125642197)

[Introduzione al sistema 3](#_Toc125642198)

[Requisiti funzionali 4](#_Toc125642199)

[Topologia del sistema 6](#_Toc125642200)

[Tool chain 6](#_Toc125642201)

[Iterazione 1 7](#_Toc125642202)

[Early Architecture 7](#_Toc125642203)

# Iterazione 0

## Introduzione al sistema

Il sistema realizzato ha lo scopo di gestire una comunità online per lo scambio di libri usati. L’idea è che se un utente ha un libro che non desidera più, può offrirlo ad altri utenti in cambio di un certo numero di **token**. Un utente intenzionato ad ottenere un libro usato dovrà “pagarlo” in token, il prezzo dipende dalla tipologia del libro, dalla sua condizione, dalla sua richiesta e disponibilità all’interno del sistema e dalla distanza che gli altri utenti devono percorrere per consegnare il libro. Infatti, la consegna dei libri viene effettuata dagli utenti stessi: ciascuno di essi indica quanti km è disposto a percorrere per trasportare i libri, e quando un libro viene acquistato il sistema identifica quali utenti (al massimo 3 incluso il “venditore”) dovranno consegnarlo e li notifica. Il “venditore”, quindi, non ottiene tutti i token versati dal “compratore” perché una porzione del prezzo viene elargita agli utenti selezionati dal sistema per la consegna (il numero di token che questi ottengono è proporzionale alla distanza che devono percorrere).

Gli utenti usufruiranno del servizio di scambio dei libri attraverso un’applicazione mobile che permetterà loro di:

* Registrarsi e gestire l’account.
* Mettere a disposizione dei libri usati.
* Comprare libri usati usando i suoi token.
* Modificare le condizioni di un libro.
* Acquistare token con valuta reale.
* Indicare i propri “libri preferiti” in modo che il sistema notifichi quando uno di tali libri è disponibile all’acquisto.

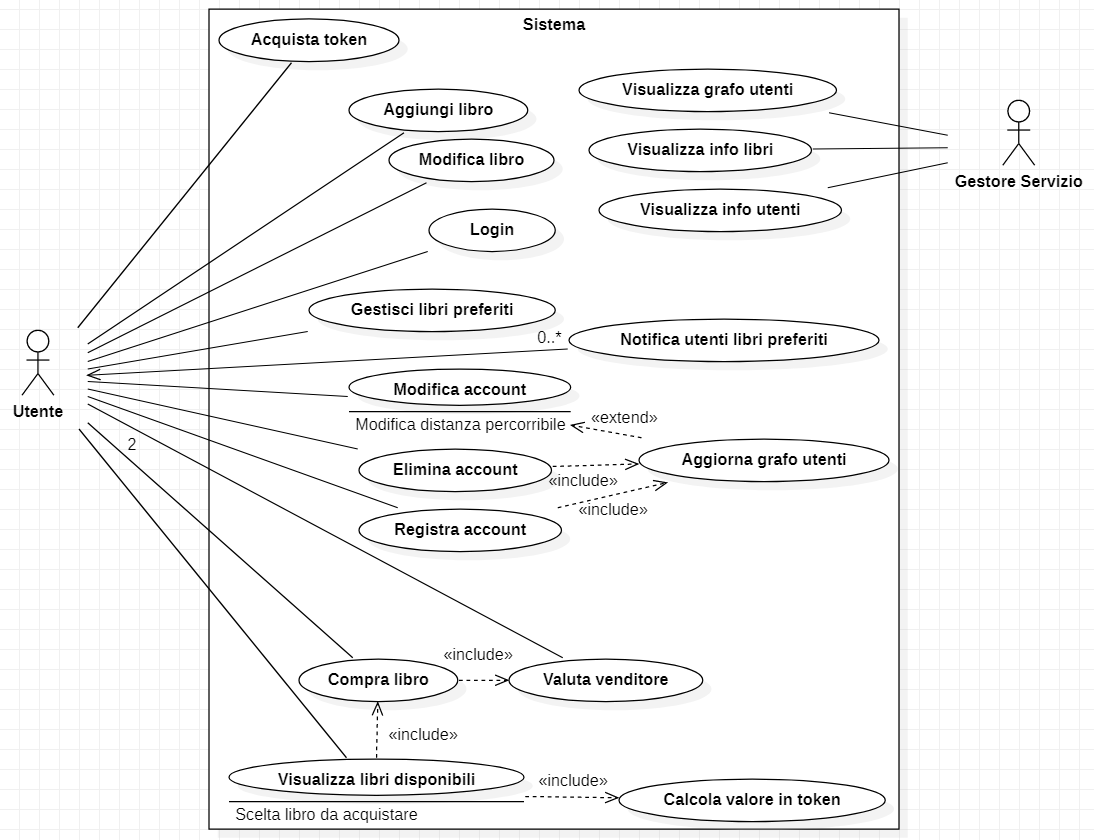
Il gestore di questo servizio si interfaccia al sistema con un’applicazione desktop dedicata che gli permette di:

* Visualizzare le statistiche sugli utenti (come chi ha più token, chi ha acquistato più libri etc.).
* Visualizzare le statistiche sui libri (quali sono i più disponibili, i più desiderati, i più scambiati etc.).
* Visualizzare il **grafo degli utenti**: ogni vertice rappresenta un utente, un arco dall’utente x all’utente y indica che l’utente y è sufficientemente vicino a x per potergli consegnare un libro. Il grafo è diretto perché se x può raggiungere y non è detto che sia vero il contrario (visto che ogni utente imposta la distanza massima che è disposto a percorrere).

Lo scopo di questo progetto non è la realizzazione del sistema nella sua interezza, verranno infatti solo implementate le componenti fondamentali seguendo la metodologia di sviluppo software AMDD (Agile Model Driven Development).

## Requisiti funzionali

I requisiti funzionali del sistema sono rappresentati nel seguente use case diagram:



In ogni iterazione del processo di sviluppo AMDD verranno scelti ed implementati alcuni di questi requisiti. Ovviamente è importante sviluppare prima i requisiti più importanti per il funzionamento del sistema, per questa ragione i requisiti sono stati divisi in cinque livelli priorità, rappresentati nella matrice che segue insieme ad altre importanti informazioni sui requisiti.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Nome – (id) | Priorità | Descrizione | | Gruppo di casi d’uso | | Fonte | Requisito Padre | | Requisiti figli | |
| Registra account (1) | Molto alta | Registrazione di un nuovo account con nome utente e password | | Gestione account | | AMDD iterazione 0 (25/01/2023) |  | | 2 | |
| Aggiorna grafo raggiungibilità (2) | Molto alta | Aggiunta di un nodo con relativi archi al grafo raggiungibilità utenti | | / | | AMDD iterazione 0 (25/01/2023) | 1, 13, 14 | |  | |
| Login (17) | Molto alta | Accesso di un utente registrato | | Gestione account | | Revisione iterazione 0 (28/01/2023) |  | |  | |
| Aggiungi libro (3) | alta | Aggiunta di un libro disponibile da parte di un utente | Gestione libri utente | | AMDD iterazione 0 (25/01/2023) | | |  | |  |
| Visualizza libri disponibili (4) | alta | Visualizzazione dei libri che possono essere ottenuti da un utente (al più distanti tre passi nel grafo raggiungibilità utenti) | Acquisto libro | | AMDD iterazione 0 (25/01/2023) | | |  | | 5, 6 |
| Calcola valore token (5) | alta | Calcolo del valore in token di un libro in funzione di che utente lo desidera | Acquisto libro | | AMDD iterazione 0 (25/01/2023) | | | 4 | |  |
| Compra libro (6) | alta | Acquisto da parte di un utente, tramite token, del libro di un altro utente | Acquisto libro | | AMDD iterazione 0 (25/01/2023) | | | 4 | | 15 |
| Visualizza grafo raggiungibilità (7) | media | Visualizzazione da parte del gestore del servizio del grafo raggiungibilità utenti | Visualizzazione informazioni di profiling | | AMDD iterazione 0 (25/01/2023) | | |  | |  |
| Visualizza info utenti (8) | media | Visualizzazione da parte del gestore di informazioni riguardanti gli utenti | Visualizzazione informazioni di profiling | | AMDD iterazione 0 (25/01/2023) | | |  | |  |
| Visualizza info libri (9) | media | Visualizzazione da parte del gestore di informazioni riguardanti i libri | Visualizzazione informazioni di profiling | | AMDD iterazione 0 (25/01/2023) | | |  | |  |
| Acquista token (10) | media | Acquisto da parte di un utente di token (con denaro reale) | Gestione account | | AMDD iterazione 0 (25/01/2023) | | |  | |  |
| Modifica libro (11) | bassa | Modifica da parte di un utente delle informazioni di un suo libro | Gestione libri utente | | AMDD iterazione 0 (25/01/2023) | | |  | |  |
| Gestisci libri preferiti (12) | bassa | Aggiunta o modifica da parte di un utente dei libri preferiti, ovvero quelli che più desidera | Gestione libri utente | | AMDD iterazione 0 (25/01/2023) | | |  | |  |
| Elimina account (13) | bassa | Eliminazione di un account | Gestione account | | AMDD iterazione 0 (25/01/2023) | | |  | | 2 |
| Modifica account (14) | bassa | Modifica delle informazioni di un utente, come posizione e massima distanza percorribile | Gestione account | | AMDD iterazione 0 (25/01/2023) | | |  | | 2 |
| Valuta venditore (15) | bassa | Valutazione di un venditore a seguito di un acquisto di libro | Acquisto libro | | AMDD iterazione 0 (25/01/2023) | | | 6 | |  |
| Notifica utenti libri preferiti (16) | Molto bassa | Notifica tramite e-mail la disponibilità di un libro tra quelli indicati come preferiti | / | | AMDD iterazione 0 (25/01/2023) | | |  | |  |

## Topologia del sistema

La topologia del sistema è rappresentata dal seguente schema in notazione libera:

Immagine che contiene tavolo

Descrizione generata automaticamente

Come si può notare la struttura è quella tipica di un sistema **Client-Server**. I Client sono di due tipi:

* L’applicazione desktop sul PC del gestore del servizio, che contatta il server tramite REST/HTTP per ricevere informazioni sui libri ed utenti presenti nel sistema
* L’applicazione Android utilizzabile dagli utenti per usufruire dei servizi offerti dal sistema. La comunicazione avviene tramite SMT/POP3 per la notifica agli utenti della disponibilità di un libro nella lista dei “libri preferiti”, il resto delle comunicazioni viene gestito tramite REST/HTTP.

Nelle comunicazioni REST/HTTP i dati vengono scambiati in formato JSON.

Lato server, sullo stesso dispositivo sono presenti sia il web server che comunica con i client sia il database che mantiene i dati sui libri ed utenti registrati.

Oltre allo stile architetturale Client-Server viene anche adottato l’approccio architetturale a **microservizi**; infatti, le funzionalità offerte dal web server non sono altro che un insieme di microservizi realizzati attraverso il framework Spring.

## Tool chain

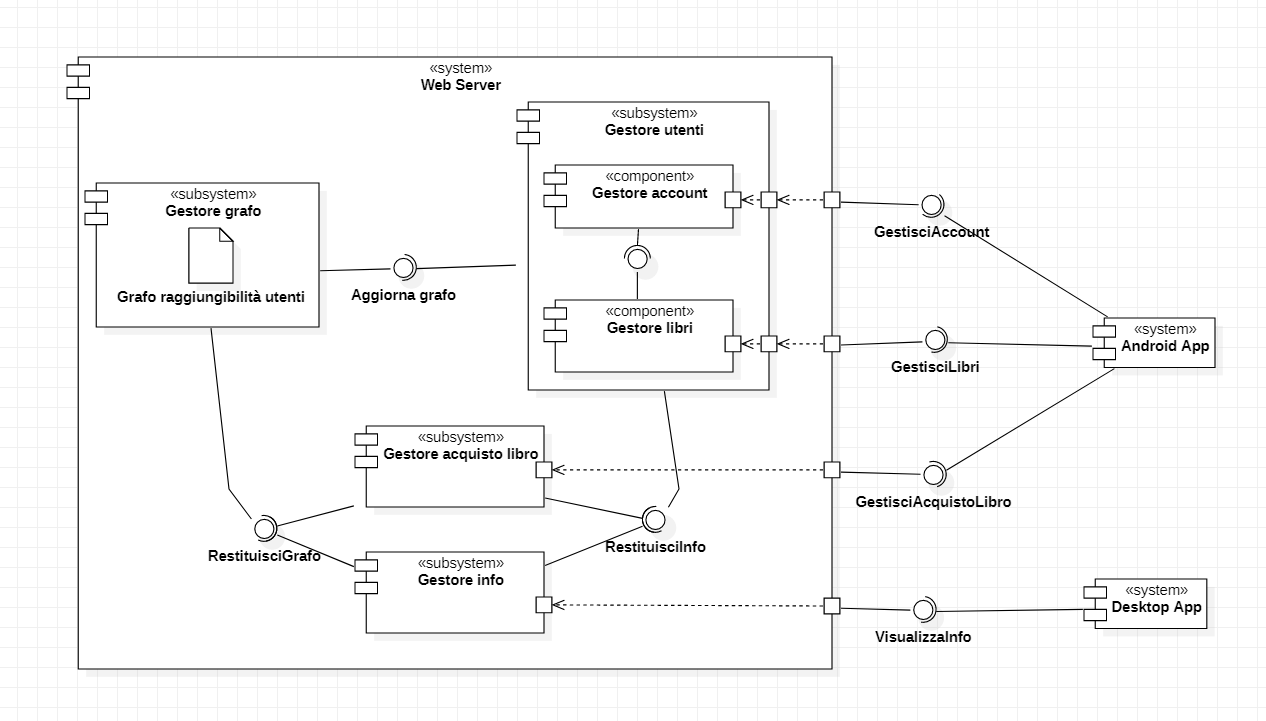
Lo sviluppo del progetto è stato supportato dai seguenti tool:

* **Eclipse**: IDE per lo sviluppo software utilizzato per la scrittura di tutto il codice (**Java**) del sistema. Alcuni degli altri tool utilizzati sono integrati con Eclipse.
* **Spring boot**: framework java per lo sviluppo di applicazioni web basate sui microservizi.
* **JAutoDoc**: plugin di Eclipse per la generazione dei Javadoc che permettono di generare la documentazione del codice java a partire dai commenti del codice.
* **JUnit 4**:framework per i test di unità in Java.
* **Eclemma**: plugin di Eclipse per la verifica della copertura del codice.
* **JGraphT**: libreria Java per la modellazione di grafi.
* **STAN4J**: software per l’analisi statica di progetti Java.
* **GitHub**: Piattaforma per il versionamento basata su Git. È stato utilizzato anche GitHub Desktop che permette di interagire dal proprio PC con GitHub utilizzando una semplice GUI.
* **StarUML**:software per la creazione di modelli UML.
* **diagrams.net**:software online per la creazione di modelli in notazione libera.

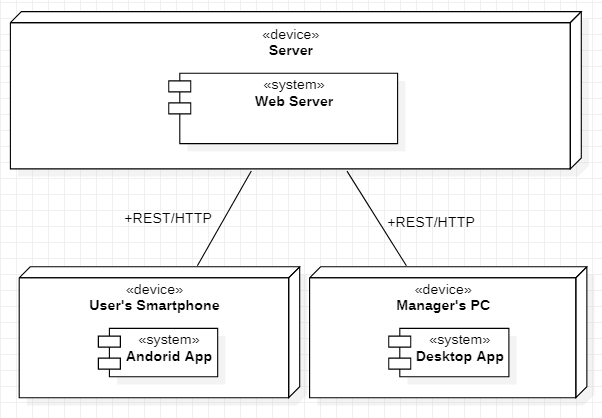
# Iterazione 1

## Early Architecture

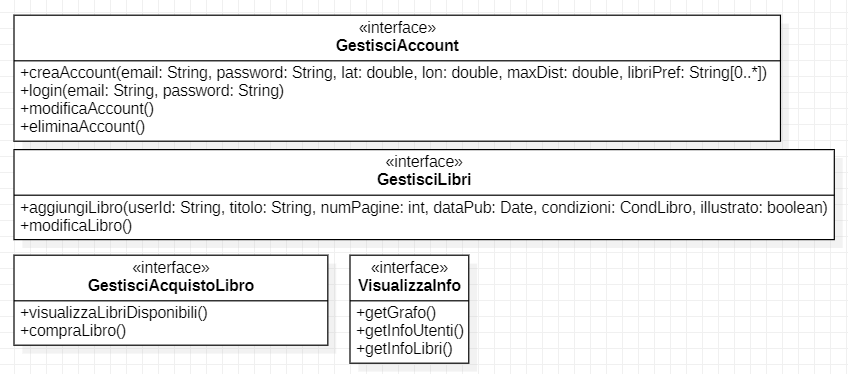
Prima della modellazione ed implementazione di un gruppo di casi d’uso, è necessario definire un’iniziale architettura del sistema ad alto livello (early architecture design). La decomposizione del sistema in sottosistemi e componenti è rappresentata nel seguente component diagram:



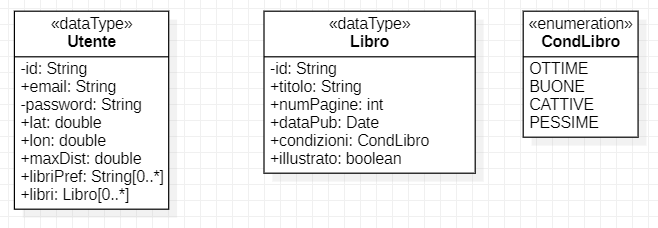
Come si può notare il focus iniziale è sulle funzionalità del web server, mentre i sistemi che rappresentano i client sono trascurati. I componenti sono mappati sui device fisici secondo il seguente deployment diagram:



Proseguendo con l’early architecture design, il seguente class diagram mostra una prima definizione delle interfacce esposte dal Web Server:



il seguente class diagram mostra i data types del sistema:



Per quanto riguarda le interfacce, sono state definite in maniera precisa solo quelle relative ai casi d’uso che sono stati implementati in questa iterazione, che vengono illustrati in seguito. Le altre sono generiche e verranno specificate nelle successive iterazioni.