**System Design Document**

1. **Introduzione**
2. **Obiettivi del sistema**

Il sistema da realizzare é un sito di dating: “UniDates.it”.

Sulla nostra piattaforma, ogni utente, puó registrarsi e creare un profilo personale caratterizzato da i propri dati anagrafici, alcuni dati personali (come sesso, altezza,interessi, colore degli occhi e dei capelli), una o piú foto personali ed una lista di **TOPIC** che esprimono alcune preferenze in diverse tematiche. Il profilo sará sempre modificabile successivamente dall’utente. Ogni utente avrá una propria sezione “Home” in cui sará possibile scorrere le foto di altri utenti che saranno selezionate dal nostro sistema di matchmaking che sceglierà quelle che piú corrispondono alle preferenze personali (Dati personali, Topic).

Se due utenti si scambiano vicendevolmente il “mi piace”, il sistema dará la possibilitá a questi ultimi di poter conversare in un’apposita finestra di chat. Un utente puó anche segnalare una foto altrui, se ritiene questa troppo offensiva, inserendo un motivo della segnalazione e specificandone i dettagli. Inoltre un utente potrá bloccare un altro utente se ritiene di essere infastidito da quest’ultimo.

La gestione del sito è affidata al Community Manager e ad i moderatori che si occuperanno di ricevere tutte le segnalazioni ricevute dagli utenti.

Il moderatore avrá il compito di inviare un ammonimento, con conseguente rimozione della foto incriminata, allo studente che non ha rispettato il regolamento del sito (specificandone motivazione e dettagli) , o nel caso di ammonimenti multipli, segnalare l’utente al CM o sospenderlo temporaneamente.

Il Community Manager (CM) avrá il compito di visualizzare gli ammonimenti inviati dai moderatori e di sospendere permanentemente un utente della piattaforma in seguito ad ammonimenti multipli.

1. **Design Goals**

**2.1 Criteri di performance**

* Il tempo di risposta per un match è di massimo 3 secondi.
* Il tempo di risposta per la ricezione di una notifica è di non più di 3 secondi.
* Il tempo di pubblicazione di una foto è di un massimo di 5 secondi.

**2.2 Criteri di affidabilità**

* Eventuali inserimenti di input non validi sono opportunamente segnalati all’utente, il quale ha la possibilità di effettuare il re-inserimento.

**2.3 Criteri di usabilità**

* L’interfaccia si adatta a qualsiasi dispositivo.

**2.4 Criteri di sicurezza**

* Tutti i dati sensibili (ad esempio password) saranno criptati per garantire la sicurezza degli utenti
* Le informazioni personali tra due studenti saranno completamente visibili soltanto quando questi due avranno un match.

1. **Trade-offs**

**[Ottimizzazione vs Portabilitá]**

Al costo di presentare una minore ottimizzazione del sistema, quest’ultimo andrá a favorire una maggiore portabilitá, potendo essere eseguito sulla maggior parte dei dispositivi, con le medesime caratteristiche.

**[Manutenzione vs Disponibilitá]**

Al costo di avere una minore disponibilità dei servizi durante il periodo di manutenzione, sarà possibile effettuare manutenzioni più veloci e dirette mirata a risolvere i problemi presenti.

**[Scalabilitá vs Costi]**

Il sistema presenterà una scalabilità elevata anche se questo dovrebbe inficiare su maggiori costi di gestione.

1. **Definizioni, acronimi ed abbreviazioni**

* RAD: Requirements Analysis Document.
* SDD: System Design Document.
* ODD: Object Design Document.
* DB: Database.
* MVC: Model View Control.
* Greenfield engineering: Tipologia di sviluppo che comincia da zero, non esiste nessun sistema a priori e i requisiti sono ottenuti dall’utente finale e dal cliente. Nasce, perciò, a partire dai bisogni dell’utente.

1. **Riferimenti**

* Bernd Bruegge & Allen H. Dutoit, Object Oriented Software Engineering: Using UML, Patterns and Java, (2nd Prentice Hall, 2003).
* <http://www.tinder.com>

1. **Architettura del sistema e dei sottoinsiemi**

**6.1 Overview**

Il modello proposto è un’applicazione web, distribuita secondo il modello client server.

L’obiettivo che si pone è di offrire un sito di incontri tra studenti universitari.

Il cliente gestisce e personalizza la parte di presentazione, il server gestisce la logica applicativa e delega le funzioni di storage ad un DBMS.

All’interno del sistema i manager sono individuati in base alle funzionalità inerenti al sottosistema, in modo tale da gestire alta coesione e basso accoppiamento.

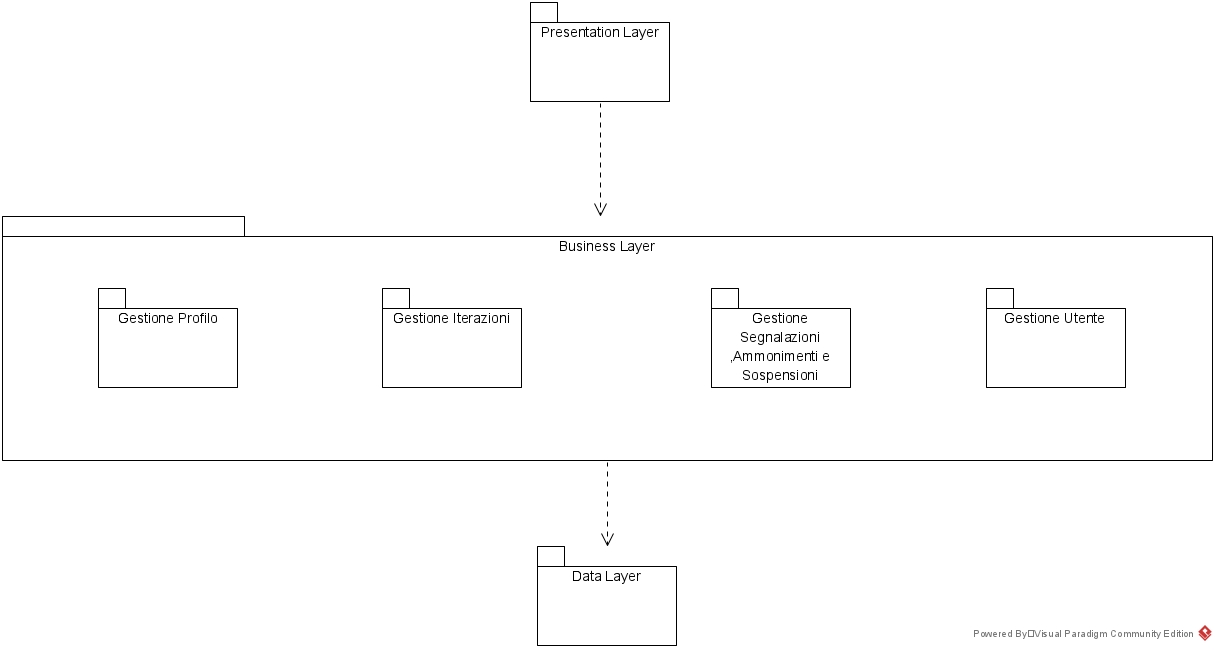
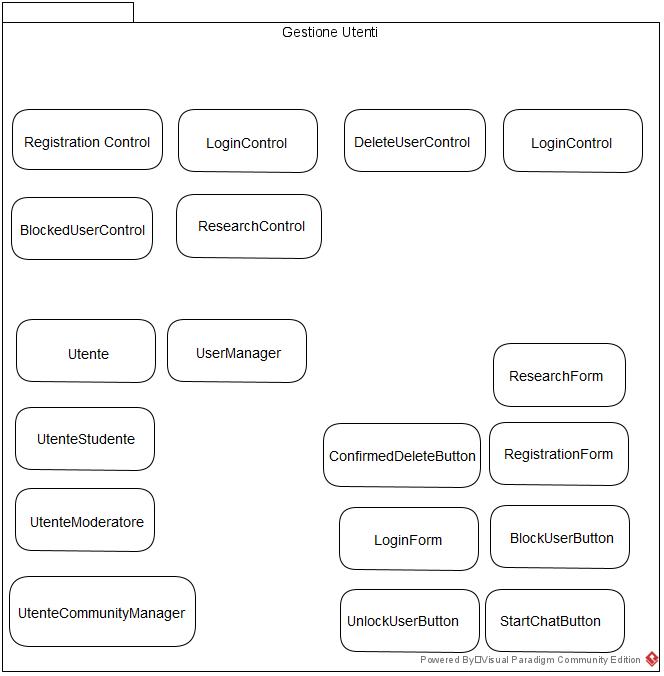
Questo aspetto rende il sistema molto più manutenibile e per questo ci siamo basati su un’architettura three tier layer.

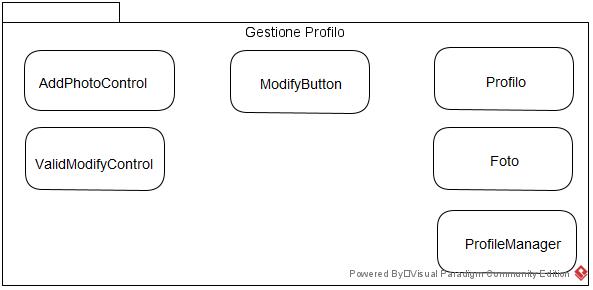
**6.2 Decomposizione in sottosistemi**

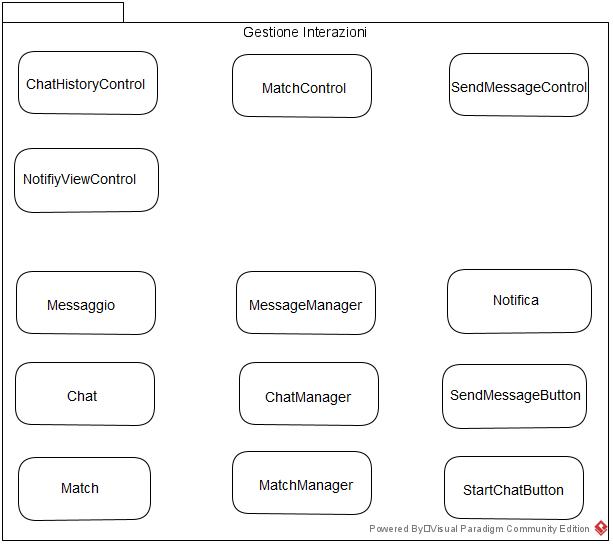
Il sistema è stato scomposto in tre sottosistemi al fine di raggiungere l’obiettivo di minimizzare l’accoppiamento e garantire una forte coesione.

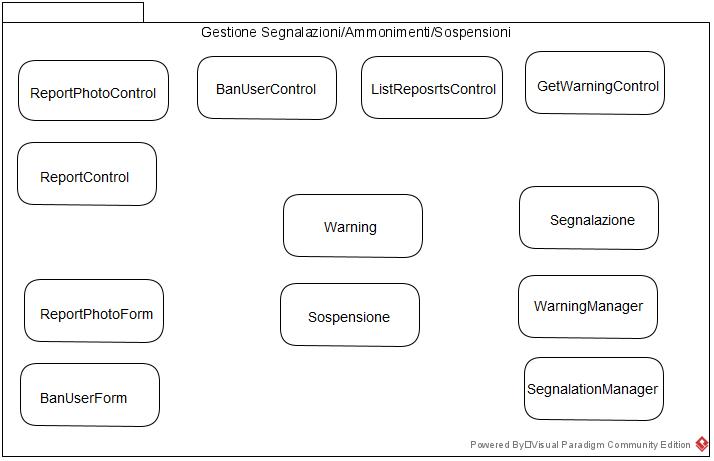
I sottosistemi sono divisi per tipo di funzionalità (dove facciamo riferimento nel documento RAD, nella sezione 1 di “Requisiti”):

* Gestione Utente: comprende tutte le funzionalità riguardanti gli utenti, e corrispondono ai requisiti funzionali: RF1, RF2, RF3,RF4, RF7, RF9, RF16, RF17,RF23.
* Gestione Interazioni: comprende tutte le funzionalitá riguardanti le interazioni tra gli utenti e lo scambio di messaggi: RF5,RF12,RF13,RF14
* Gestione Profilo: comprende tutte le funzionalità che rendono possibile la cancellazione e la modifica del profilo. Le funzionalità corrispondono ai requisiti funzionali:RF6, RF8, RF10, RF11.
* Gestione Segnalazioni ed Ammonimenti e Sospensioni: comprende le funzionalità di segnalazioni, notifiche di ammonimento e sospensioni/rimozione utente e/o foto utente. Le funzionalità corrispondono ai requisiti: RF15, RF18, RF19, RF20, RF21, RF22, RF24.

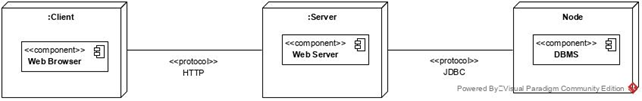








**6.3 Mapping Hardware-Software**



Il sistema che si vuole realizzare sarà installato su una macchina che fungerà da server, sarà quindi provvista di un Web Server (verrà utilizzato Apache Tomcat). Il server interagirà tramite JDBC con un database (verrà utilizzato H2). Il client sarà un qualsiasi personal computer provvisto di browser ed una connessione ad internet.

1. **Gestione persistente dei dati**

Per la persistenza dei dati, si è pensato di utilizzare un database piuttosto che un file, la scelta è basata sul fatto si necessita di un’organizzazione logica dei dati, di sicurezza per l’utilizzo e gestione dei dati sensibili.Inoltre la scelta è influenzata dal fatto che i dati sono accessibili in concorrenza da più utenti che li richiedono.

I dati che saranno resi persistenti sono (Riferimenti RAD, sezione: “Class Diagram”):

* Studente, Moderatore, Community Manager
* Notifica
* Match
* Chat
* Messaggio
* Profilo
* Foto
* Segnalazione
* Ammonimento

1. **Gestione e controllo della sicurezza**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attori/Sottosistemi** | **Gestione Utente** | **Gestione Interazioni** | **Gestione Profilo** | **Gestione segnalazioni/ammonimenti/sospensioni** |
| **Utente non registrato** | * Registrazione |  |  |  |
| **Studente** | * Login * Bloccare/Sbloccare uno studente * Ricerca utente * Cancellare il account | * Mettere “Mi piace” * Iniziare una chat * Visualizzare l’archivio delle chat * Visualizzare foto altrui * Visualizzare le notifiche | * Aggiungere/Rimuovere foto * Modificare le informazioni del profilo * Visualizzare profili altrui | * Segnalare una foto * Visualizzare ammonimenti ricevuti |
| **Moderatore** | * Login * Ricerca utente | * Iniziare una chat di moderazione * Visualizzare l’archivio delle chat | * Visualizzare profili altrui | * Inviare un ammonimento * Sospendere temporaneamente uno studente * Visualizzare segnalazioni utente |
| **Community Manager** | * Login * Ricerca utente | * Iniziare una chat di moderazione * Visualizzare l’archivio delle chat | * Visualizzare profili altrui | * Inviare un ammonimento * Sospendere uno studente * Visualizzare segnalazioni utente * Visualizzare ammonimenti inviati |

1. **Boundary Condition**

|  |  |
| --- | --- |
| ***Nome*** | **Startup del sistema** |
| ***Attori partecipanti*** | Amministratore del sistema. |
| ***Entry condition*** | L’amministratore del sistema si trova sulla macchina che ospita il web server. |
| ***Flusso di eventi*** | 1. L’amministratore avvia il sistema  2. Il sistema inizia l’esecuzione e  inizializza i manager |
| ***Exit condition*** | Il sito diventa raggiungibile. |

|  |  |
| --- | --- |
| ***Nome*** | **Shutdown del sistema** |
| ***Attori partecipanti*** | Amministratore del sistema. |
| ***Entry condition*** | L’amministratore del sistema si trova sulla macchina che ospita il web server che è attualmente in esecuzione. |
| ***Flusso di eventi*** | 1. L’amministratore sospende il sistema.  2. Il sistema notifica gli utenti dello shutdown.  3. Il sistema termina l’esecuzione. |
| ***Exit condition*** | Il sito non è più raggiungibile. |

1. **Servizi dei sottosistemi**
2. **Gestione utenti**

|  |  |
| --- | --- |
| **Servizi** | **Descrizione** |
| registrazione | Servizio che si occupa della registrazione di un utente non registrato al sistema. |
| login | Servizio che permette ad un utente registrato di effettuare il login al sistema. |
| bloccaStudente | Servizio che permette ad uno studente di bloccare uno studente. |
| sbloccaStudente | Servizio che permette ad uno studente di sbloccare uno studente bloccato |
| ricercaUtente | Servizio che permette ad un utente di ricercare gli altri utenti tramite indirizzo email |
| esisteUtente | Servizio che permette di controllare se un utente é presente nel sistema |

1. **Gestione profilo**

|  |  |
| --- | --- |
| **Servizi** | **Descrizione** |
| aggiungiFoto | Servizio che permette ad uno studente di aggiungere una foto al profilo. |
| rimuoviFoto | Servizio che permette ad uno studente di eliminare una foto dal profilo. |
| modificaProfilo | Servizio che permette allo studente di modificare le informazioni personali. |
| visualizzaProfilo | Servizio che permette ad un utente di visualizzare sia il proprio profilo personale sia quello di altri studenti. |

1. **Gestione interazioni**

|  |  |
| --- | --- |
| **Servizi** | **Descrizione** |
| avviareChat | Servizio che permette ad un utente di avviare una chat. |
| visualizzaNotifiche | Servizio che permette ad un utente di visualizzare tutte le notifiche. |
| effettuaMatch | Servizio di sistema che crea il match tra due studenti. |
| invioMessaggio | Servizio che consente ad un utente di inviare un messaggio ad un altro utente. |

1. **Gestione Segnalazioni/Ammonimenti/Sospensioni**

|  |  |
| --- | --- |
| **Servizi** | **Descrizione** |
| segnalaFoto | Servizio che permette di segnalare una foto |
| ammonimentiRicevuti | Servizio che permette di visualizzare ammonimenti ricevuti |
| inviaAmmonimento | Servizio che permette di inviare un ammonimento |
| sospensioneUtente | Servizio che permette di sospendere uno studente |
| segnalazioniUtente | Servizio che permette di visualizzare le segnalazioni utente |
| ammonimentiInviati | Servizio che permette di visualizzare ammonimenti inviati |