



# Facoltà di Ingegneria - Università di Catania

## Smart Network University Communications

## Software Engineering Projects

Relatore

**Prof. Orazio Tomarchio**

Studenti

**Russo Leandro**  
**Invincibile Daniele**  
**Didomenico Nicola**

Dipartimento di Ingegneria Elettrica, Elettronica e Informatica – DIEEI  
Facoltà di Ingegneria Informatica

27 Febbraio 2015



## Indice

### 1. Indice

#### 1.1. Prefazione

### 2. Ideazione - Iterazione 1

#### 2.1. Iterazione 1: Requisiti

#### 2.2. Iterazione 1: Analisi - UC1\_RequestConnection

#### 2.3. Iterazione 1: Analisi - UC2\_AccessRoom

#### 2.4. Iterazione 1: Progettazione - UC1\_RequestConnection

#### 2.5. Iterazione 1: Progettazione - UC1\_AccessRoom

#### 2.6. Iterazione 1: Implementazione - UC1 e UC2

#### 2.7. Iterazione 1: Test - UC1 e UC2

### 3. Refactoring Iterazione 1

#### 3.1. Refactoring Iterazione 1 - UC1\_RequestConnection

#### 3.2. Refactoring Iterazione 1 - UC1\_AccessRoom



## Indice

- 3.3. Refactoring Iterazione 1: Implementazione - UC1 e UC2
- 3.4. Refactoring Iterazione 1: Test - UC1 e UC2

## 4. Elaborazione - Iterazione 2

- 4.1. Iterazione 2: Requisiti - UC3\_SendPublicMessage
- 4.2. Iterazione 2: Analisi - UC3\_SendPublicMessage
- 4.3. Iterazione 2: Progettazione - UC3\_SendPublicMessage
- 4.4. Iterazione 2: Implementazione - UC3\_SendPublicMessage
- 4.5. Iterazione 2: Test - UC3\_SendPublicMessage

## 5. Elaborazione - Iterazione 3

- 5.1. Iterazione 3: Requisiti - UC4\_SendPrivateMessage
- 5.2. Iterazione 3: Analisi - UC4\_SendPrivateMessage
- 5.3. Iterazione 3: Progettazione - UC4\_SendPrivateMessage
- 5.4. Iterazione 3: Implementazione - UC4\_SendPrivateMessage



## Indice

### 5.5. Iterazione 3: Implementazione - UC4\_SendPrivateMessage

## 6. Riferimenti



## Metodologia Applicata

Per lo sviluppo del processo software descritto a breve, è stata applicata la metodologia dell'Unifiled Process (UP) suggerita durante il corso di studi di Ingegneria del Software.

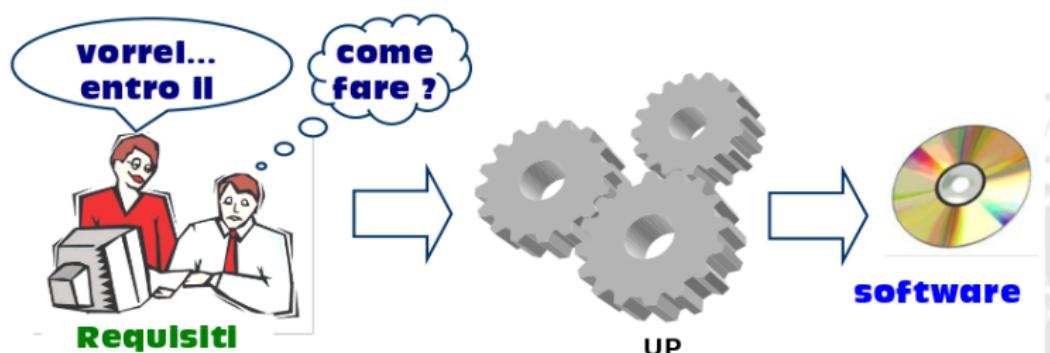
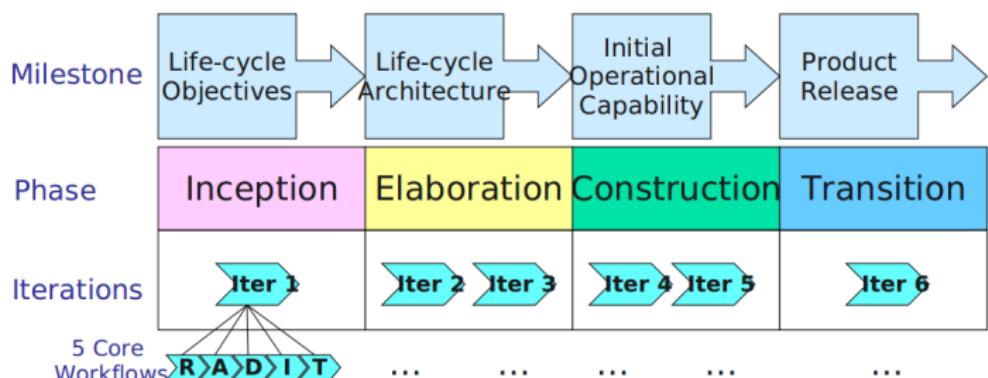


Figura 1 : Software Engineering Process



## Struttura di UP

Il ciclo di vita del progetto è diviso in quattro fasi: Principio (Inception), Elaborazione (Elaboration), Costruzione (Construction) e Transizione (Transition), come si può vedere nella figura 2.



**Figura 2 :** Fasi dell'Unified Process

## Struttura di UP

Ogni fase è formata da una o più iterazioni. Per ogni iterazione si sviluppano in modo iterativo e incrementale 5 attività di lavoro (workflows) RADIT: Requisiti (Requirements), Analisi (Analys), Progettazione (Design), Implementazione (Implementation)e Test, come si può vedere nella figura 3.

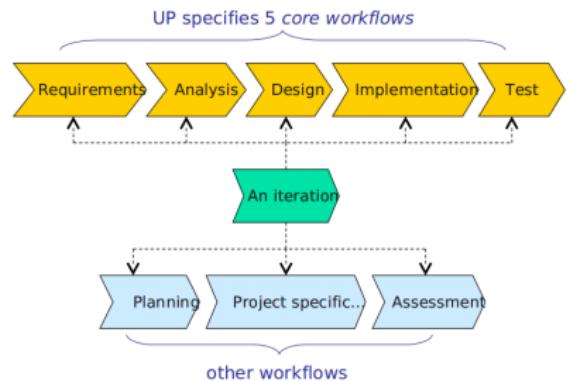


Figura 3 : Iterazioni dell'Unfiled Process



## Documenti costitutivi del progetto

### *INSERIRE DESCRIZIONE*

- 
- 
- 





## Descrizione realtà d'interesse

Nel Piano Nazionale della Ricerca (PNR) messo a punto da HORIZON ITALIA e MIUR viene bandito un progetto chiamato:

**“Smart Network University Communications (SNUC)”** destinato a tutti gli atenei italiani.

Tale progetto è caratterizzato dalle seguenti specifiche descritte nel seguito, che consente ad appassionati, studenti, ricercatori, docenti e imprese di scambiarsi messaggi in tempo reale su condivisioni, integrazioni e competenze di idee per una contaminazione tra ambiti disciplinari diversi, realtà diverse e stimolare nei partecipanti lo sviluppo della cultura dell'intraprendere e dell'innovazione.

In tale sistema si richiedono le seguenti specifiche:



## Descrizione realtà d'interesse

- Esistono diversi canali o stanze virtuali (ciascuna legata a un corso di laurea per ogni facoltà, includendo sia la triennale e la magistrale di quel determinato corso di laurea) nelle quali un utente può entrare per scambiare messaggi con gli altri utenti presenti nella stessa stanza.
- È possibile inviare due tipi di messaggi:
  - ▶ “pubblici”: dei messaggi inviati da un utente e trasmessi a tutti gli altri partecipanti presenti nella stanza;
  - ▶ “privati”: dei messaggi inviati da un utente e trasmessi ad uno specifico partecipante presente nella stessa stanza, in questo caso il destinatario selezionato sarà l'unico a ricevere il messaggio.
- In ogni istante il sistema prevede la presenza di una tipologia di utente particolare, chiamato amministratore. I compiti più importanti di un amministratore sono i seguenti:



## Descrizione realtà d'interesse

- ▶ può creare o eliminare stanze del servizio di messaggistica;
  - ▶ inviare messaggi di avviso ad un utente;
  - ▶ in caso di comportamenti irregolari è possibile espellere un utente dalla stanza e l'utente non può più rientrare fin quando questo non sarà rimosso dalla lista dei partecipanti bannati della stanza presente nel sistema.
- ▶ Il sistema deve mandare dei messaggi di notifica, che permettono di aggiornare l'utente di un cambiamento dello stato del sistema.



## Iterazione 1: Requisiti - Documento di Visione

La caratteristica principale del progetto è quella di consentire ad appassionati, studenti, ricercatori, docenti e imprese di scambiarsi messaggi in tempo reale per ogni ateneo. In particolare in questo progetto sono presenti diverse stanze virtuali che rappresentano le facoltà di un ateneo con due tipologie di utilizzatori di sistema, ovvero l'amministratore e gli utenti del servizio di messaggistica. Con le seguenti caratteristiche:

- l'amministratore gestisce le stanze virtuali e supervisiona gli utenti del servizio;



## Iterazione 1: Requisiti - Documento di Visione

- gli utenti per usufruire di tale servizio inseriscono un nickname e si collegano al server impostando dei parametri di connessione. Ogni utente può accedere a una stanza e inviare messaggi ad ogni utente presente nella stanza in tempo reale, inoltre ognuno può contattare in maniera privata gli altri partecipanti presenti nella stanza.

L'architettura utilizzata per offrire il servizio è di tipo client-server, questo approccio consente di fornire un'interfaccia più flessibile per l'accesso al servizio di messaggistica anche con un semplice browser, senza modificare pesantemente la progettazione rispetto ad un architettura peer-to-peer.

## Iterazione 1: Requisiti - Diagrammi dei casi d'uso

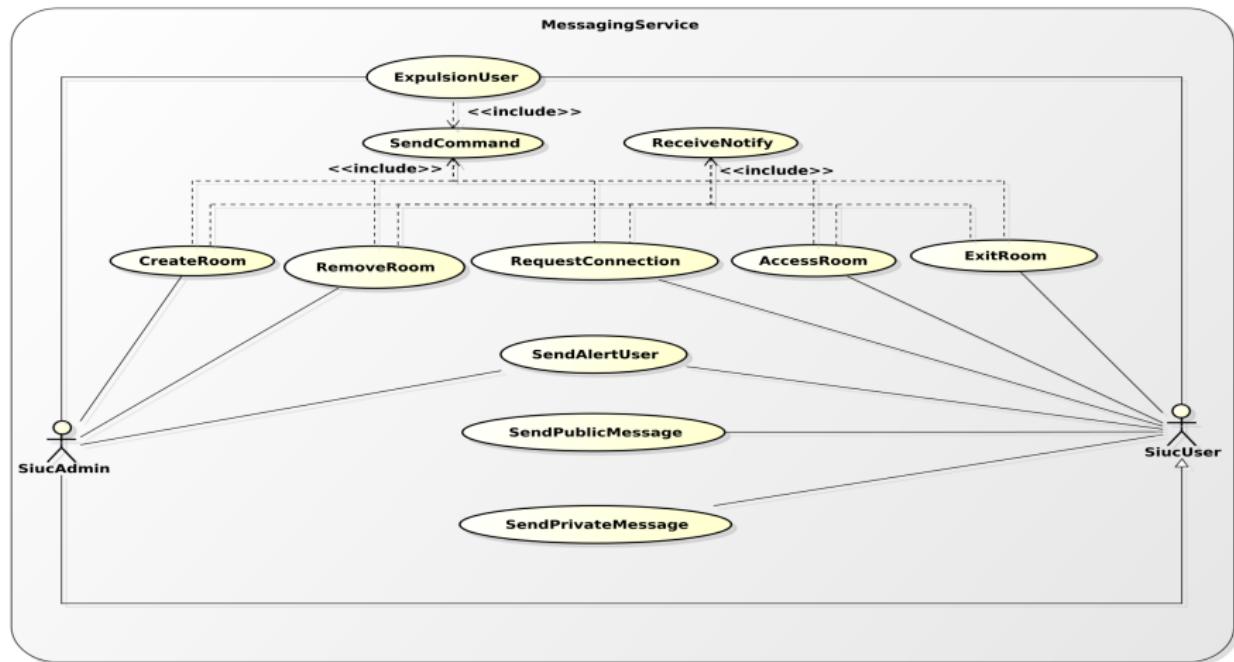


Figura 4 : Diagrammi dei casi uso



## Iterazione 1: Requisiti - Modello dei casi d'uso

Attori Identificati: Amministratore (SnucAdmin) e Utente del Servizio di Messaggistica (SnucUser).

Casi d'uso identificati: RequestConnection, AccessRoom, SendPublicMessage, SendPrivateMessage, ReceiveNotify, SendCommand, ExitRoom, CreateRoom, RemoveRoom, SendAlertUser, ExpulsionUser.

Descrizione breve dei casi d'uso identificati:

- ① RequestConnection. L'utente si connette al sistema specificando il nickname, l'indirizzo e la porta del servizio di messaggistica.
- ② AccessRoom. L'utente richiede al sistema l'ingresso in una specifica stanza del servizio (che deve essere stata precedentemente creata dall'amministratore del servizio).



## Iterazione 1: Requisiti - Modello dei casi d'uso

- ③ SendPublicMessage. Il partecipante al servizio di messaggistica invia un messaggio “pubblico” che viene inviato dal sistema a tutti i partecipanti che si trovano nella stessa stanza di colui che ha inviato il messaggio.
- ④ SendPrivateMessage. Il partecipante al servizio di messaggistica invia un messaggio “privato” ad uno specifico partecipante.
- ⑤ ReceiveNotify. Il sistema manda dei messaggi di notifica, che permettono di aggiornare l’utente di un cambiamento dello stato del sistema.
- ⑥ SendCommand. ....



## Iterazione 1: Requisiti - Modello dei casi d'uso

- ⑦ ExitRoom. Un partecipante richiede al sistema l'uscita dal servizio. Questo comporta la sua eliminazione dall'insieme dei partecipanti presenti nella stanza del servizio cui era precedentemente associato l'utente.
- ⑧ CreateRoom. L'amministratore del servizio di messaggistica è responsabile della creazione (preventiva) delle stanze che potranno successivamente essere visitate dai partecipanti.
- ⑨ RemoveRoom. In ogni momento l'amministratore può eliminare una stanza dal servizio (ad esempio perchè non ci sono partecipanti). Ciò comporta l'invio preventivo di un messaggio a tutti i partecipanti eventualmente presenti nella stanza, i quali potranno in seguito richiedere l'ingresso in una nuova stanza del servizio.



## Iterazione 1: Requisiti - Modello dei casi d'uso

- ⑩ SendAlertUser. L'amministratore può in ogni momento inviare un messaggio di avviso ad uno specifico partecipante al servizio.
- ⑪ ExpulsionUser. L'amministratore può in ogni momento espellere da una stanza uno specifico partecipante.





## Glossario

- ▶ SnucUser, User, Utente: è l'utente del servizio di messaggistica che usufruisce dei servizi forniti.
- ▶ SnucAdmin, Admin, Amministratore: è il gestore del servizio di messaggistica, che possiede la facoltà di creare e/o eliminare una stanza, di inviare un particolare messaggio di avviso ad un partecipante per un comportamento non corretto ed eventualmente di bannarlo eliminandolo dalla stanza.
- ▶ Messaging Service, Servizio di messaggistica, sistema, chat: sistema che si occupa della gestione.
- ▶ Room, stanza, canale: insieme che si può differenziare dalle altre in base alle tematiche trattate.



## Glossario

- ▶ Messaggio Pubblico: messaggio testuale inviato da un partecipante a tutti gli utenti presenti nella stanza.
- ▶ Messaggio Privato: messaggio testuale inviato da un partecipante ad un particolare partecipante presente nella stanza.
- ▶ Notifica: particolari messaggi di avviso inviati dal server agli utenti del servizio.





## Iterazione 1: Requisiti, caso d'uso UC1\_RequestConnection

**Tabella 1 :** Descrizione dettagliata: caso d'uso UC1\_RequestConnection

Nome caso d'uso	UC1_RequestConnection
Portata	Applicazione Smart Intelligent University Communications
Livello	Obiettivo Utente
Attore primario	SnucUser
Parti interessate e interessi	SnucUser: vuole collegarsi al servizio di messaggistica
Pre-condizioni	L'utente ha bisogno di una connessione di rete
Post-condizioni (garanzia di successo)	L'utente è inserito tra gli utenti online con il nickname confermato dal servizio di messaggistica ottenendo un messaggio di benvenuto
Scenario principale di successo	<ul style="list-style-type: none"> <li>① L'utente inserisci un nickname, l'address e la porta del server.</li> <li>② Il sistema esamina il nickname inviato dall'utente e verifica se è presente una omonimia.</li> <li>③ Il sistema conferma l'inserimento tra gli utenti online inviando una notifica di benvenuto.</li> </ul>



## Iterazione 1: Requisiti, caso d'uso UC1\_RequestConnection

Estensioni (o flussi alternativi)	<p>1A - SnucUser inserisce parametri errati:</p> <ul style="list-style-type: none"> <li>▶ Viene visualizzato un messaggio di errore e viene richiesto nuovamente l'inserimento di tali parametri.</li> </ul> <p>2A - Omonimia del nickname:</p> <ul style="list-style-type: none"> <li>▶ Il sistema cambia il nickname aggiungendo _ al nickname (es. _nickname).</li> <li>▶ Il sistema conferma l'inserimento tra gli utenti online inviando una notifica di benvenuto.</li> </ul>
Requisiti speciali (Requisiti Non Funzionali)	Comunicazione asincrona in cui lo scambio di informazioni avviene in tempo reale, senza sensibili pause tra invio e ricezione del messaggio.
Elenco delle varianti tecnologiche	L'applicazione dovrebbe essere flessibile al funzionamento di diversi protocolli di comunicazione (es. TCP, UDP) e con diversi strati middleware (es. Socket, RMI)
Frequenza di ripetizione	Potrebbe essere quasi ininterrotta
Varie e/o Problemi Aperti	//



## Iterazione 1: Requisiti, caso d'uso UC2\_AccessRoom

**Tabella 2 :** Caso d'uso UC2\_AccessRoom

Nome caso d'uso	UC2_AccessRoom
Portata	Applicazione Smart Intelligent University Communications
Livello	Obiettivo Utente
Attore primario	SnucUser
Parti interessate e interessi	SnucUser: vuole registrarsi e effettuare l'ingresso in una stanza presente nel servizio di messaggistica
Pre-condizioni	Nel sistema è presente almeno una stanza creata da un amministratore.
Post-condizioni (garanzia di successo)	Nel caso di svolgimento normale l'utente è registrato ed è presente nell'insieme degli utenti della stanza specificata.



## Iterazione 1: Requisiti, caso d'uso UC2\_AccessRoom

Scenario principale di successo	<ul style="list-style-type: none"> <li>① L'utente richiede una lista di stanze presenti nel sistema di messaggistica.</li> <li>② Il sistema invia la lista delle stanze.</li> <li>③ L'utente seleziona la stanza tra quelle presenti in lista.</li> <li>④ Il sistema registra l'utente alla stanza.</li> <li>⑤ Il sistema visualizza a video gli utenti presenti nella stanza.</li> <li>⑥ Il sistema visualizza un'area pubblica dove vengono mostrati tutte le conversazioni in corso dal quel momento in poi.</li> </ul>
Estensioni (o flussi alternativi)	3A - Il SnucUser inserisce una stanza non in elenco: <ul style="list-style-type: none"> <li>① Il sistema invia un messaggio di errore.</li> </ul>
Requisiti speciali (Requisiti Non Funzionali)	Comunicazione asincrona in cui lo scambio di informazioni avviene in tempo reale, senza sensibili pause tra invio e ricezione del messaggio.
Elenco delle varianti tecnologiche	L'applicazione dovrebbe essere flessibile al funzionamento di diversi protocolli di comunicazione (es. TCP, UDP) e con diversi strati middleware (es. Socket, RMI)
Frequenza di ripetizione	Potrebbe essere quasi ininterrotta
Varie e/o Problemi Aperti	//



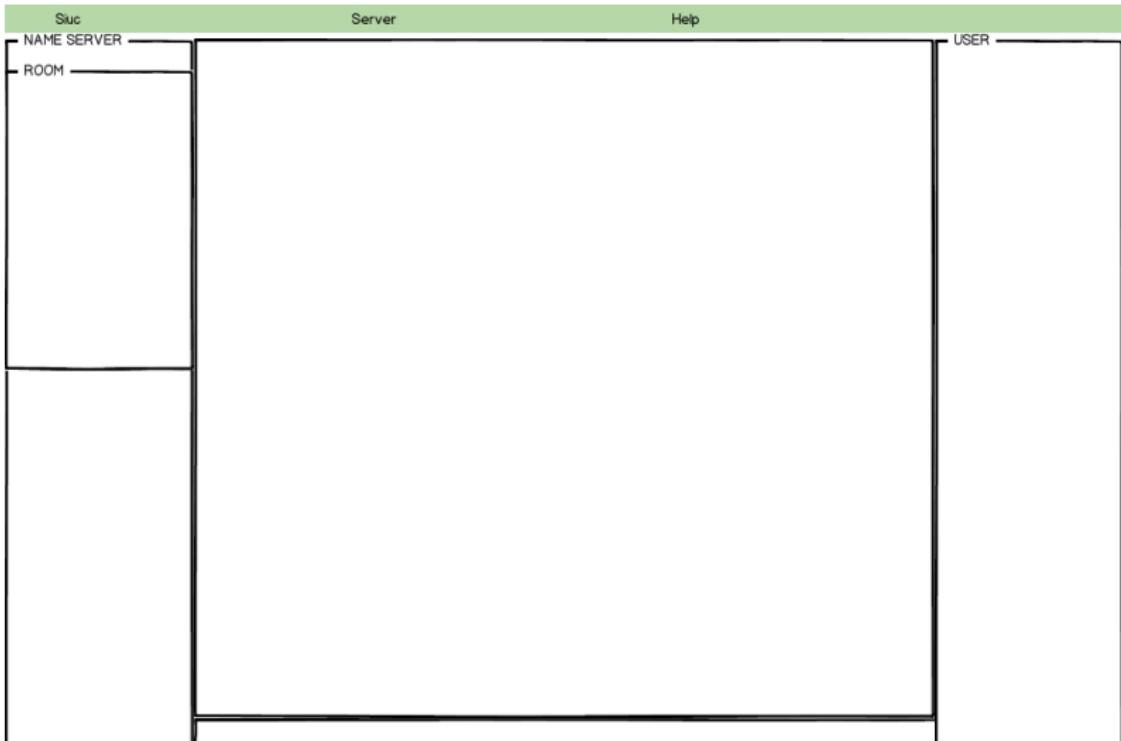
# Descrizione Mockups User Interface

*INSERIRE DESCRIZIONE*



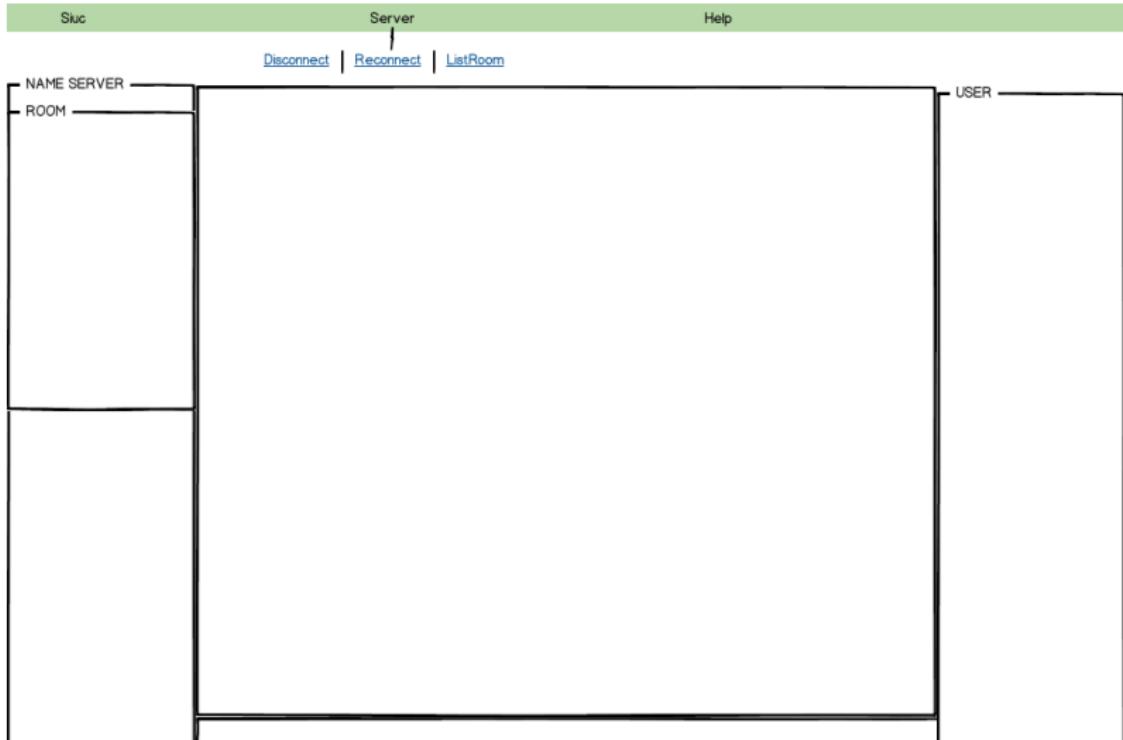


# Iterazione 1: Requisiti - Mockups Bozza User Interface



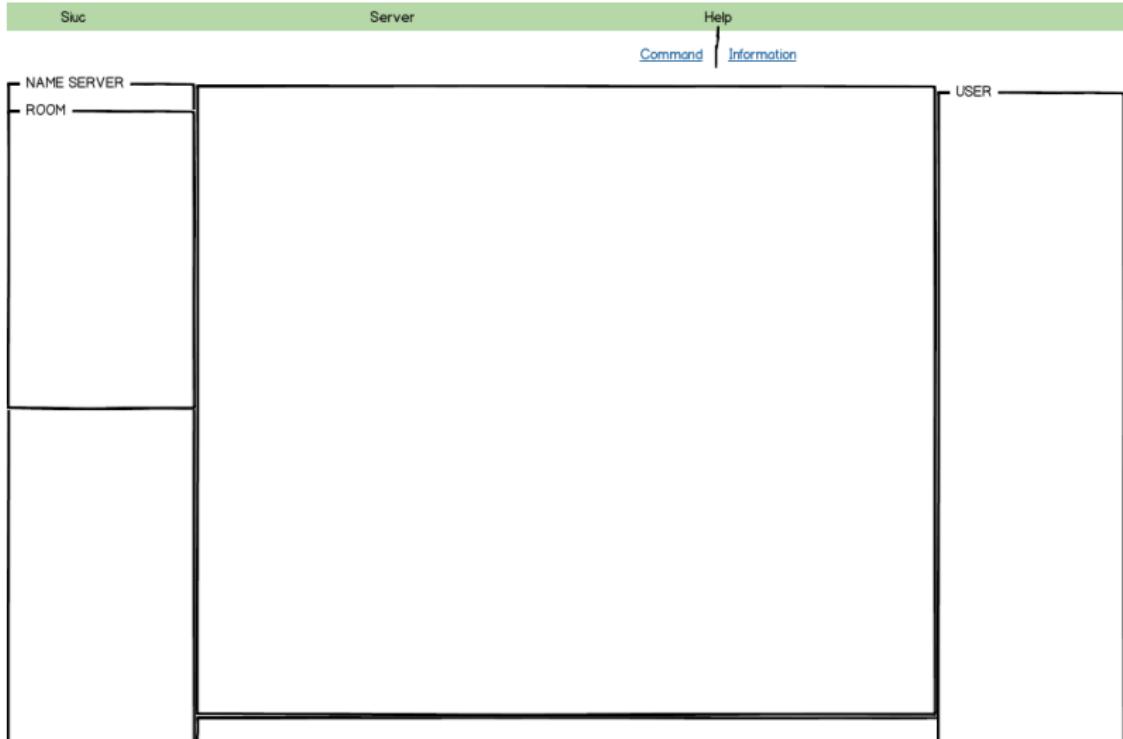


## Iterazione 1: Requisiti - Mockups Bozza User Interface



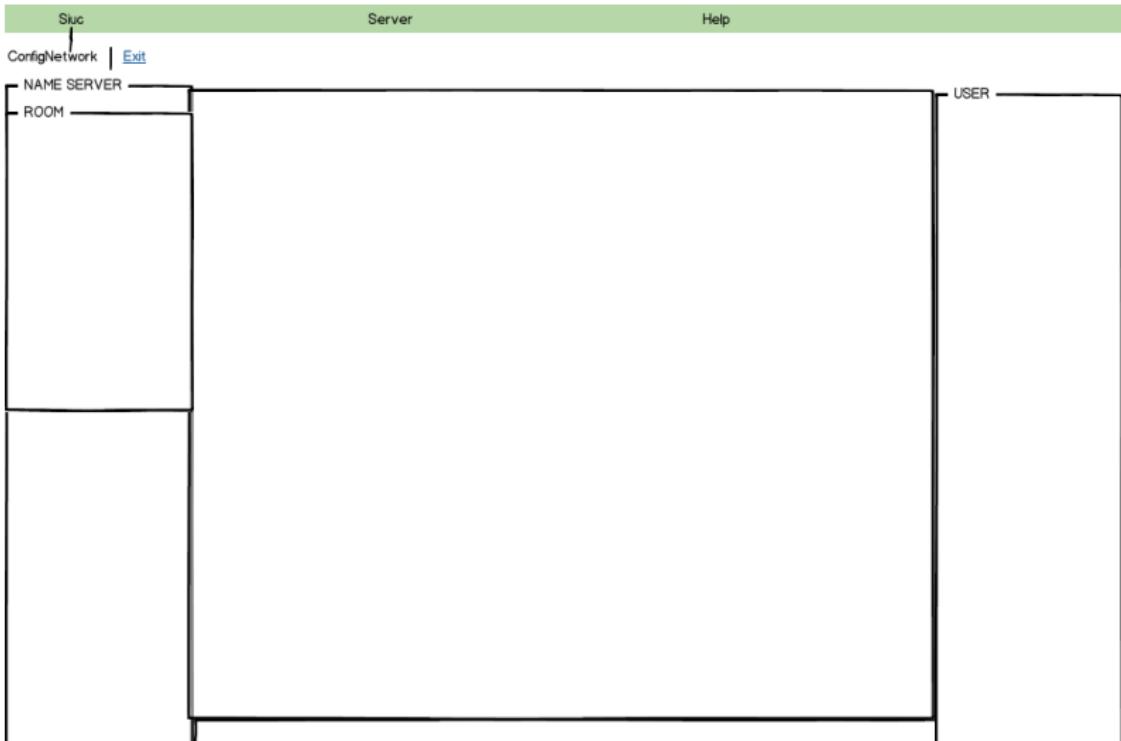


# Iterazione 1: Requisiti - Mockups Bozza User Interface



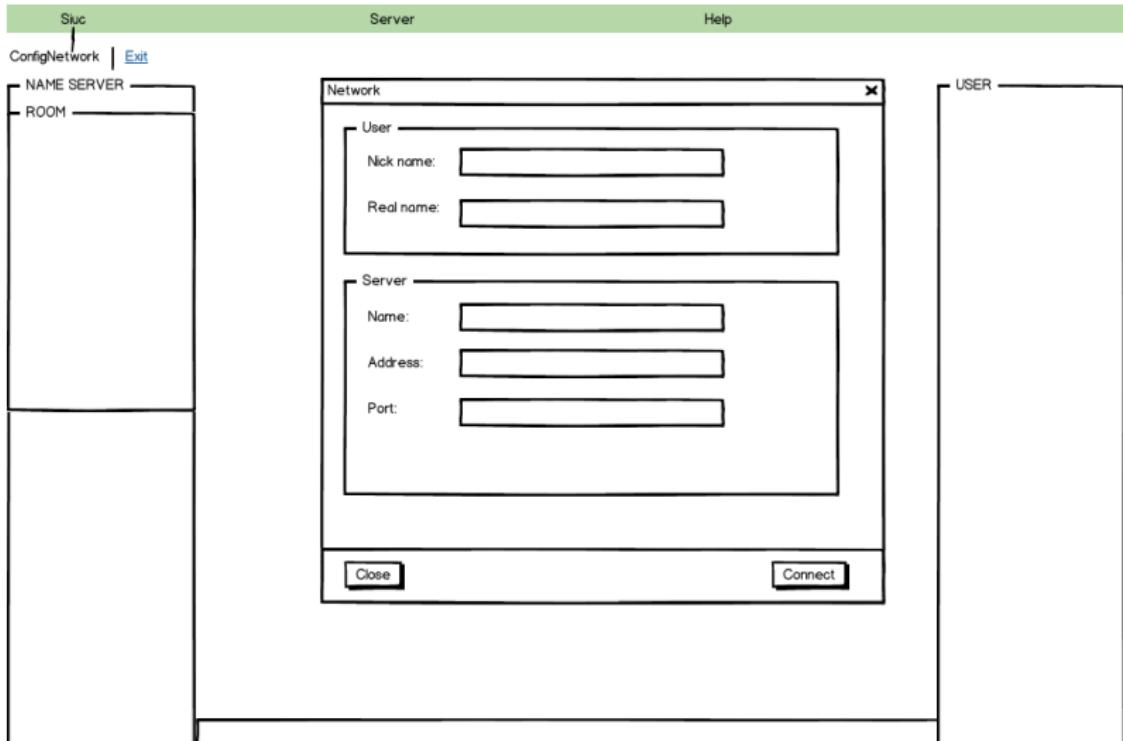


# Iterazione 1: Requisiti, Mockups Bozza User Interface



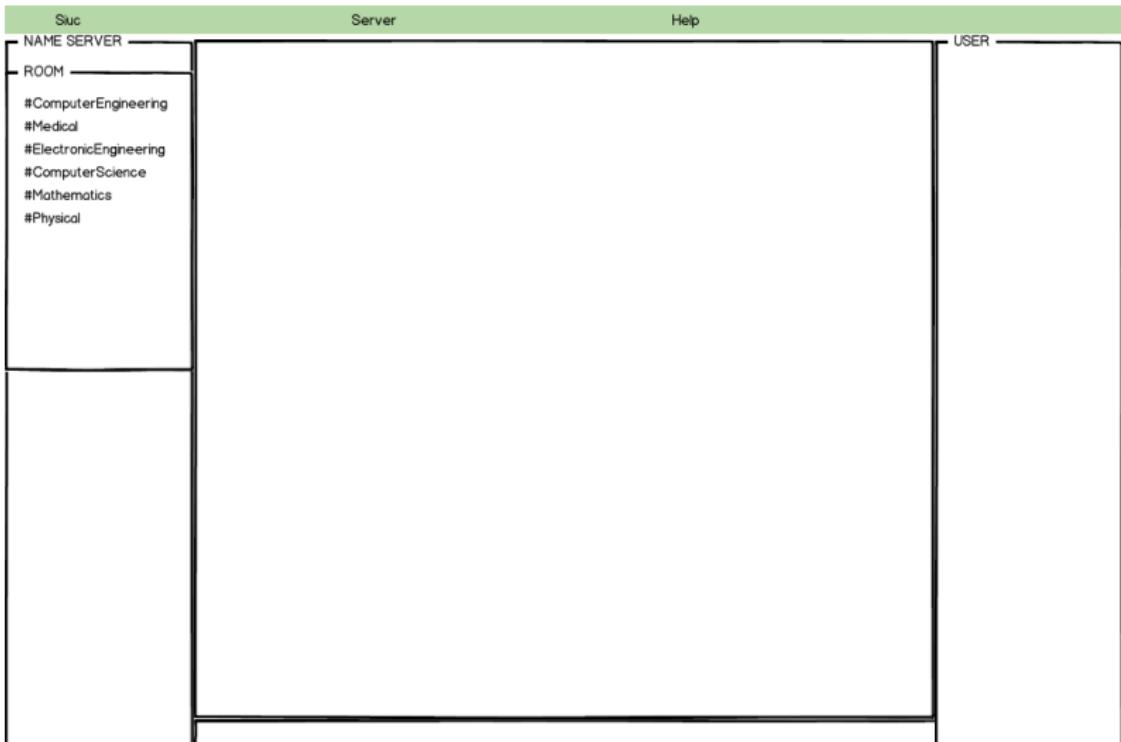


## Iterazione 1: Requisiti, Mockups Bozza User Interface





# Iterazione 1: Requisiti, Mockups Bozza User Interface





# Iterazione 1: Requisiti, Mockups Bozza User Interface

Suc	Server	Help	USER
NAME SERVER ROOM #ComputerEngineering #Medical #ElectronicEngineering #ComputerScience #Mathematics #Physical	19 Gennaio 2015 <p>[ora:minuti:secondi] nicola : ciao</p> <p>[ora:minuti:secondi] leandro : funziona ?</p> <p>[ora:minuti:secondi] daniele : bellissima .....</p> <p>[ora:minuti:secondi] orazio : è un ottimo modello UP...</p>		o nicola - leandro - daniele - orazio



# Iterazione 1: Requisiti, Mockups Bozza User Interface

The mockup shows a window titled "Private Chat" with the date "04 Febbraio 2015". Inside the window, there is a list of messages:

- [ora:minuti:secondi] nicola : ciao
- [ora:minuti:secondi] leandro : ciao
- [ora:minuti:secondi] nicola : riusciremo nell'impresa ?
- [ora:minuti:secondi] leandro : cettu, cettu, w sant'agata !!!
- [ora:minuti:secondi] nicola : no w san michele !!!

The interface has a green header bar with tabs for "Sicu", "Server", and "Help". On the left, a sidebar lists "NAME SERVER" and "ROOM" categories with various channels like "#ComputerEngineering" highlighted. On the right, a sidebar lists "USER" accounts: o nicola, leandro, daniele, and orazio.



# Iterazione 1: Requisiti, Mockups Bozza User Interface

Sluc	Server	Help
NAME SERVER ROOM #ComputerEngineering <b>#Medical</b> #ElectronicEngineering #ComputerScience #Mathematics #Physical	<p align="center"><b>19 Gennaio 2015</b></p> <p> <a href="#">[ora:minuti:secondi]</a> user_medical 1 : ragazzi questi informatici ne sanno una più del dia  <a href="#">[ora:minuti:secondi]</a> user_medical 2 : si...  <a href="#">[ora:minuti:secondi]</a> user_medical 3 : assurdo ....  <a href="#">[ora:minuti:secondi]</a> user_medical 4 : oggi c'è la lezione di anatomia ??         </p>	USER <ul style="list-style-type: none"> <li>o user_medical 1</li> <li>- user_medical 2</li> <li>- user_medical 3</li> <li>- user_medical 4</li> </ul>



## Descrizione Analisi - UC1\_RequestConnection

*INSERIRE DESCRIZIONE*





## Iterazione 1: Analisi - UC1\_RequestConnection

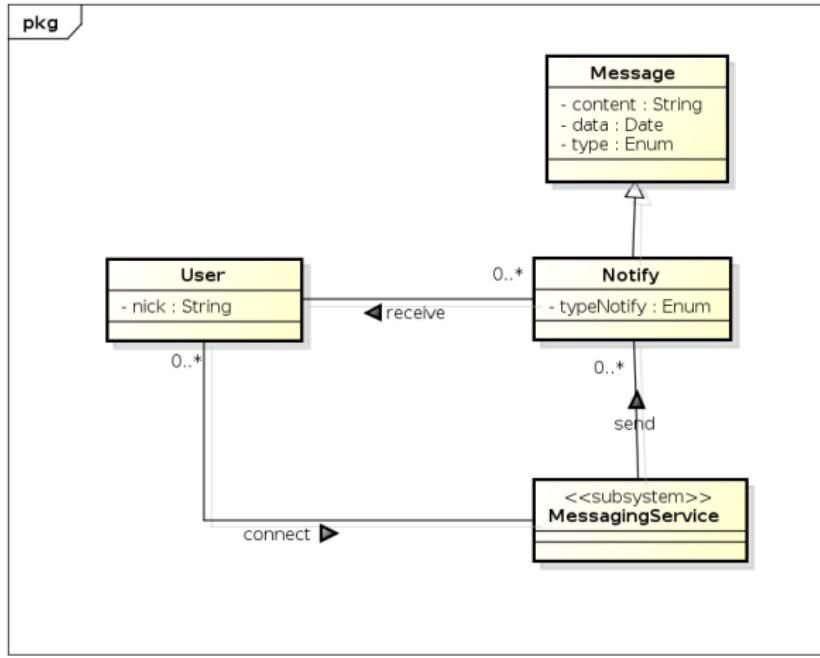


Figura 5 - UC1 - Modello di dominio

## Iterazione 1: Analisi - UC1\_RequestConnection

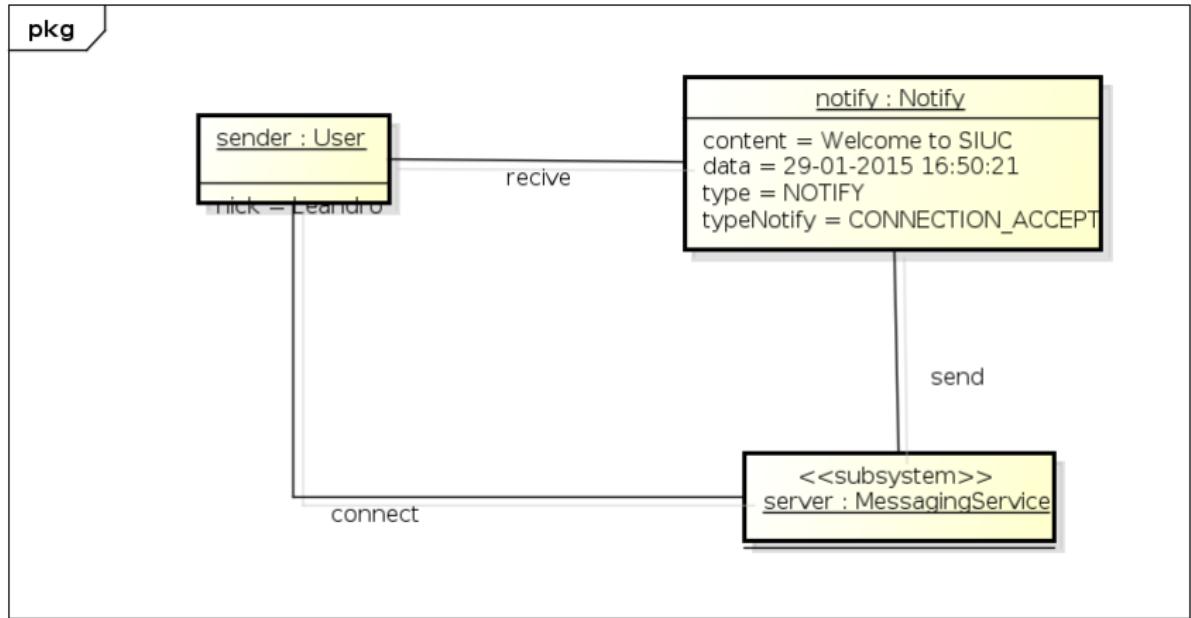


Figura 6 : UC1 - Oggetti di dominio

## Iterazione 1: Analisi - UC1\_RequestConnection

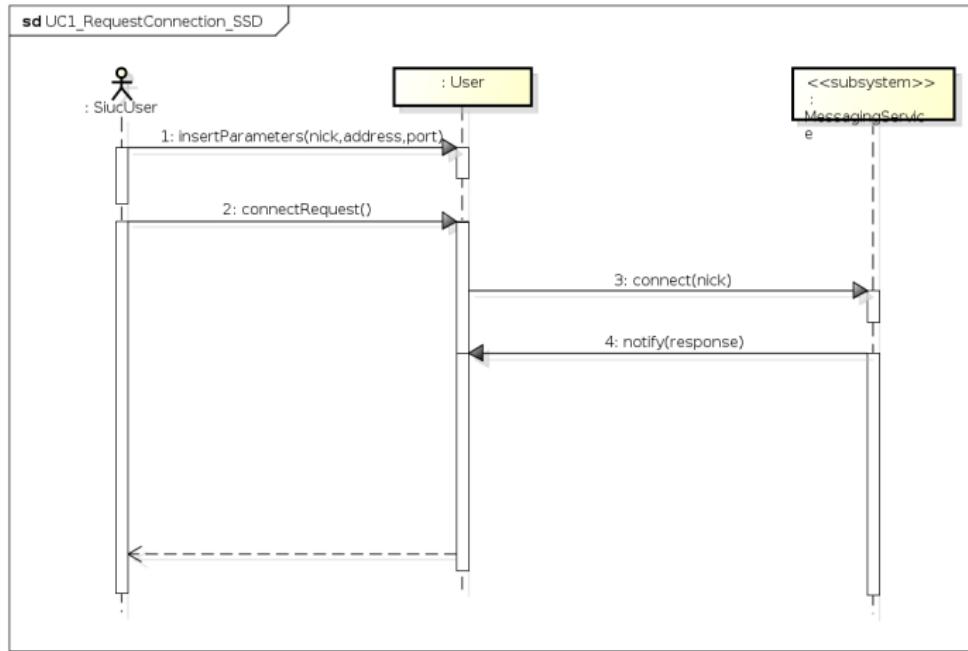


Figura 7 : UC1 - Diagramma di sequenza di sistema



# Iterazione 1: Analisi, UC1 contratto CO1 - NameCO1

**Tabella 3 : UC1 Contratto CO1 - NameCO1**

Operazione	
Riferimenti	
Pre-condizione	
Post-condizione	





## Descrizione Analisi - UC2\_AccessRoom

*INSERIRE DESCRIZIONE*



## Iterazione 1: Analisi - UC2\_AccessRoom

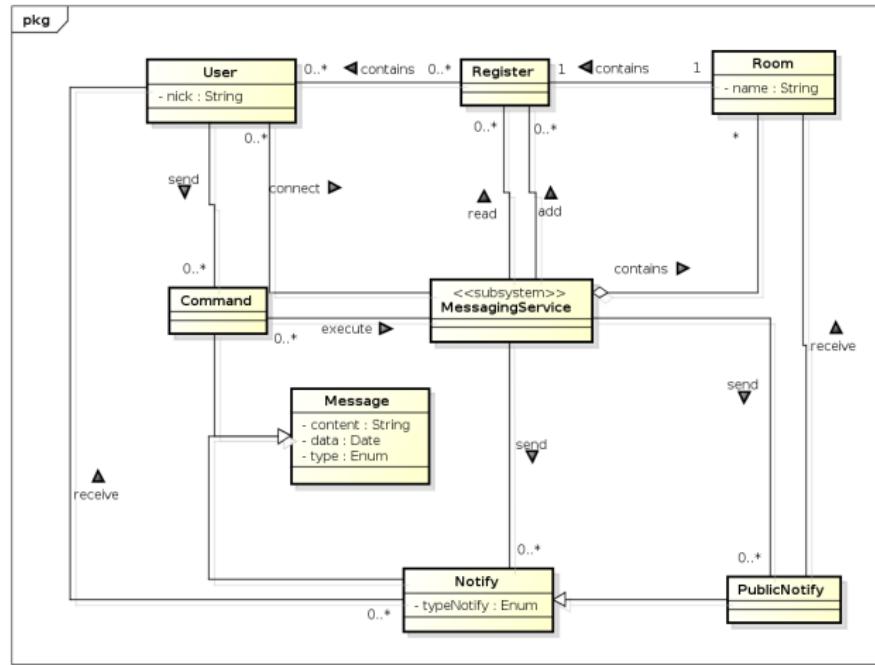


Figure 9 - UC2 - Modello di iterazione 1

## Iterazione 1: Analisi - UC2\_AccessRoom

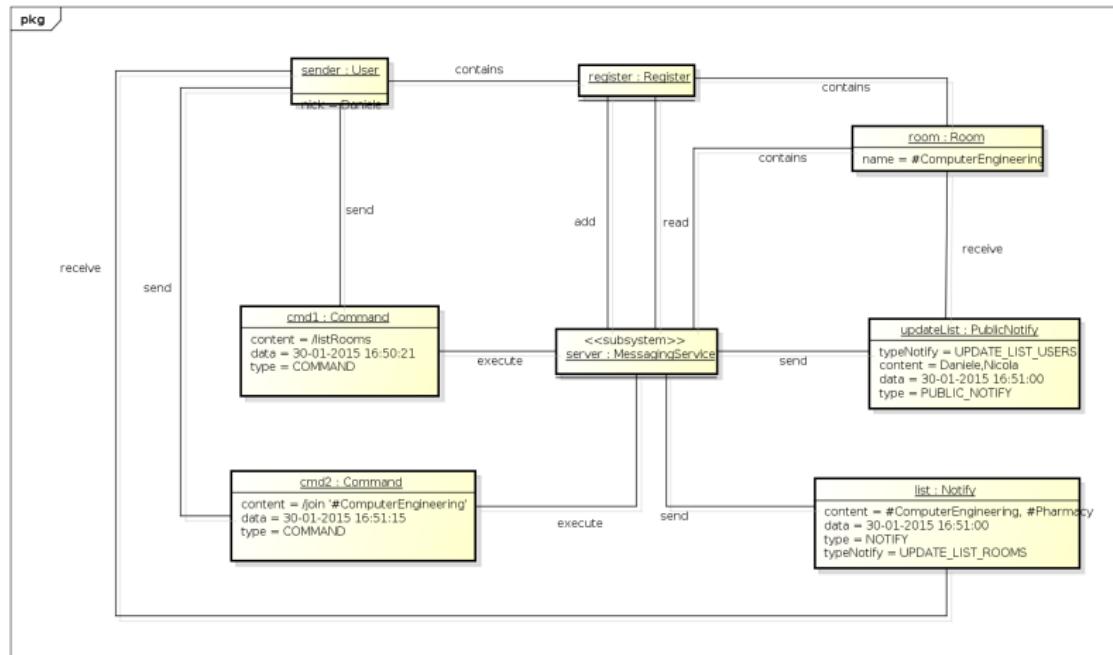
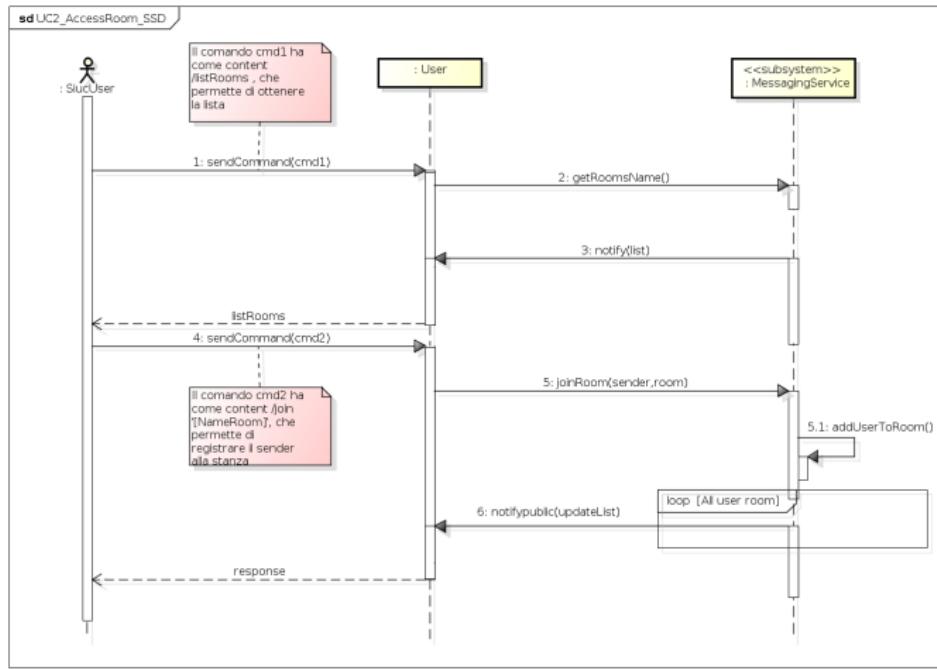


Figura 9 : UC2 - Oggetti di dominio

## Iterazione 1: Analisi - UC2\_AccessRoom





# Iterazione 1: Analisi, UC2 contratto CO2 - NameCO2

**Tabella 4 :** UC2 Contratto CO2 - NameCO2

Operazione	
Riferimenti	
Pre-condizione	
Post-condizione	





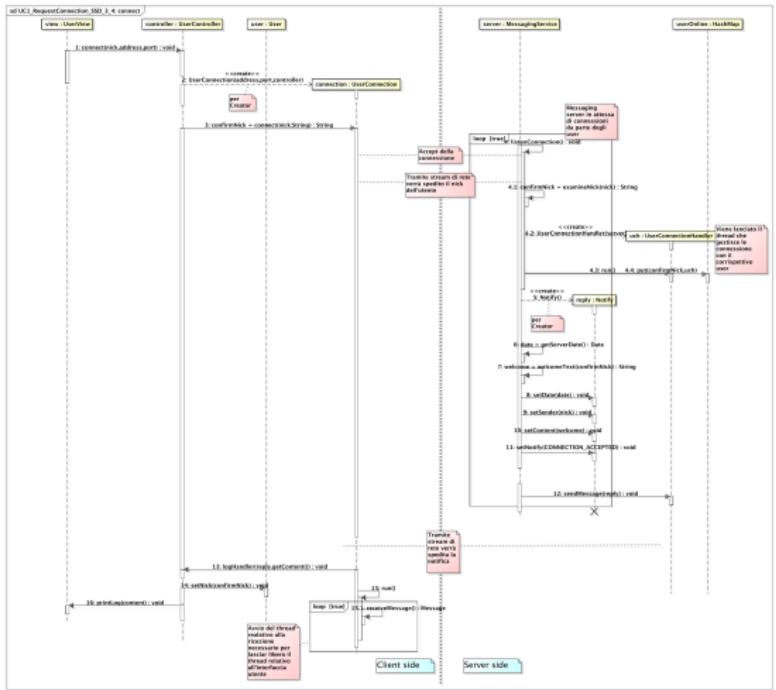
# Descrizione Progettazione - UC1\_RequestConnection

*INSERIRE DESCRIZIONE*





## Iterazione 1 Progettazione: UC1\_RequestConnection - OP3-4





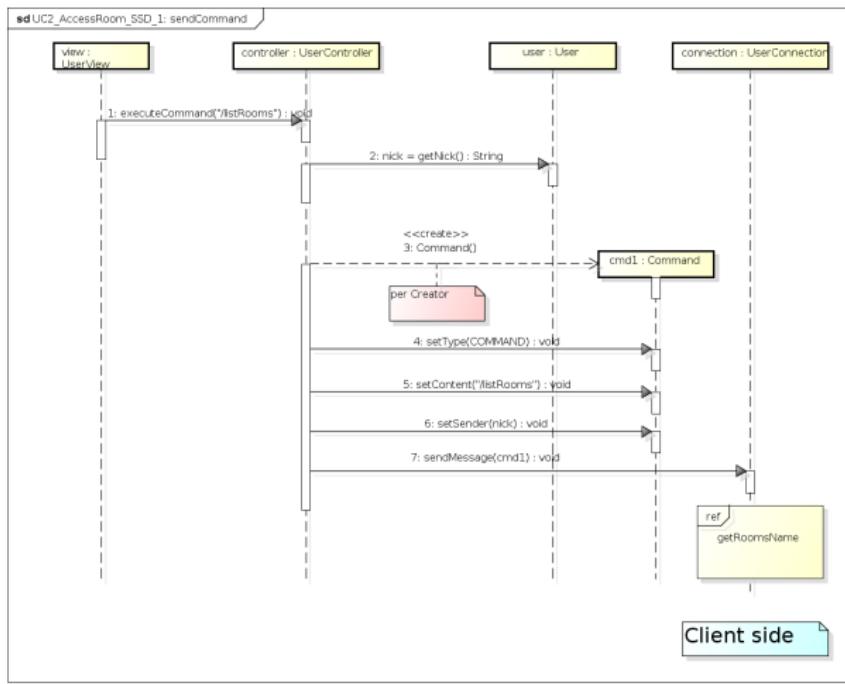
# Descrizione Progettazione - UC2\_AccessRoom

*INSERIRE DESCRIZIONE*

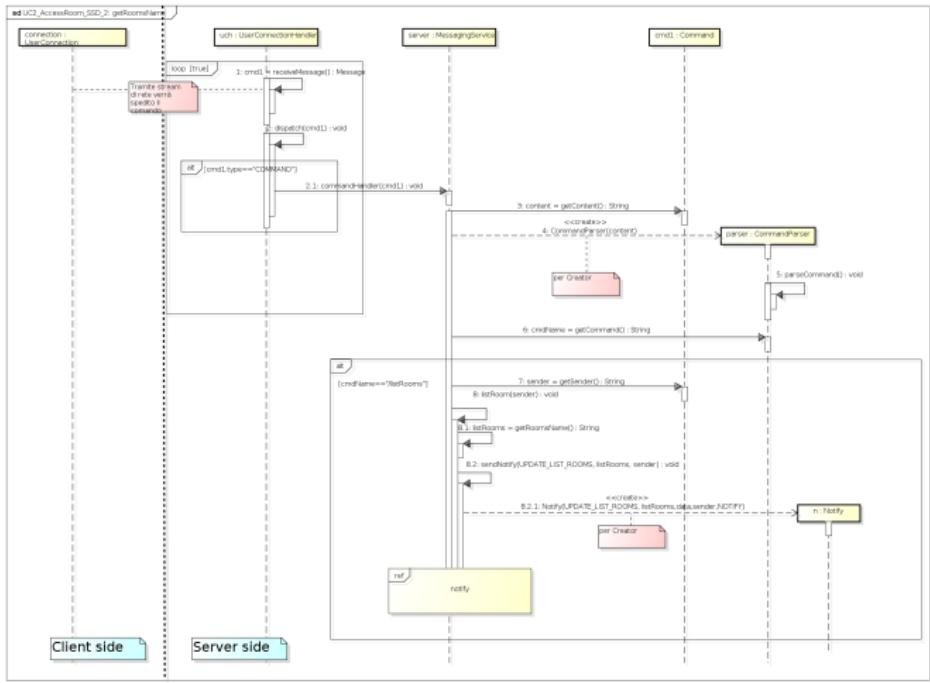




# Iterazione 1 Progettazione: UC2\_AccessRoom - OP1



## Iterazione 1 Progettazione: UC2\_AccessRoom - OP2



## Iterazione 1 Progettazione: UC2\_AccessRoom - OP3

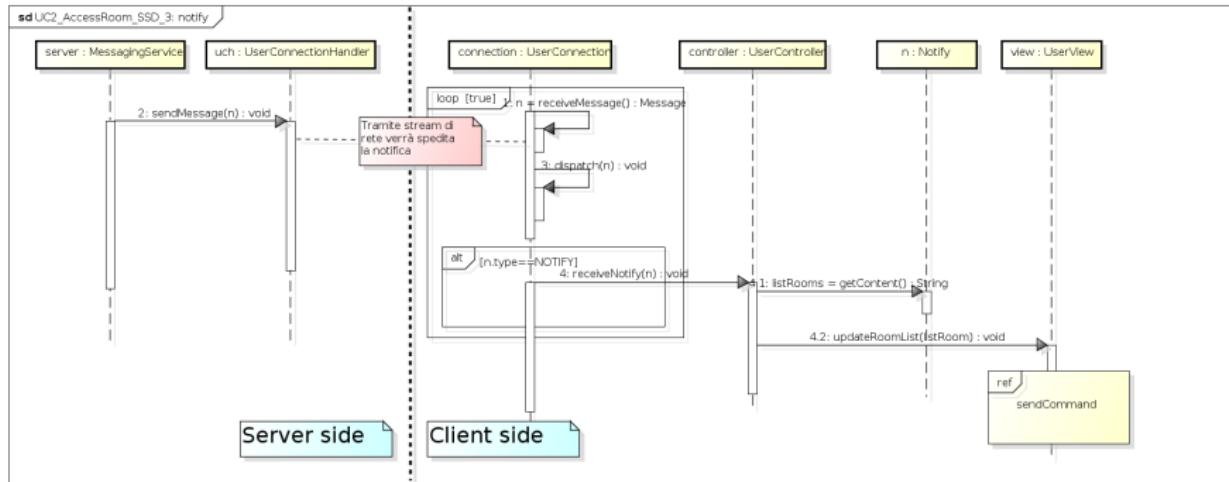
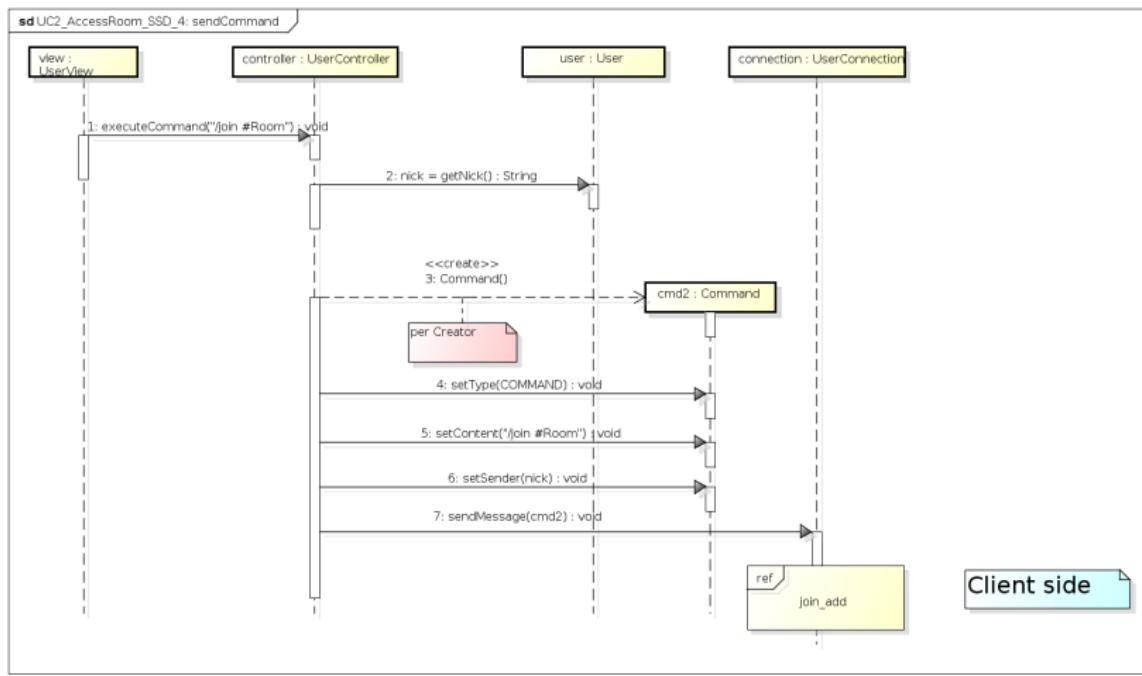


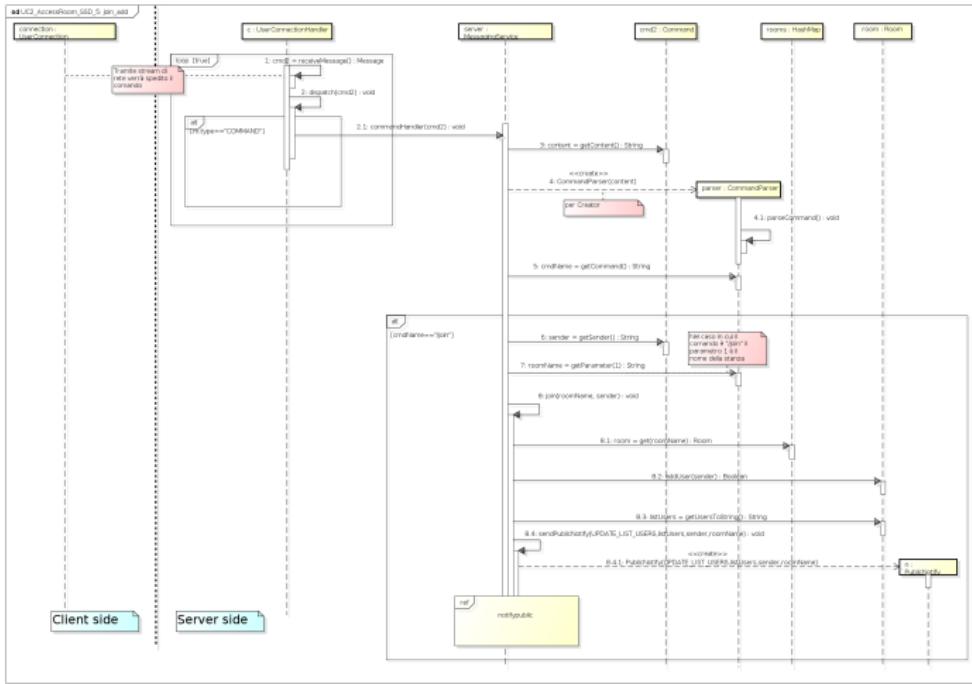
Figura 14 : SSD - OP3: notify(list) del modello di dominio (figura 10)



# Iterazione 1 Progettazione: UC2\_AccessRoom - OP4



## Iterazione 1 Progettazione: UC2\_AccessRoom - OP5



## Iterazione 1 Progettazione: UC2\_AccessRoom - OP6

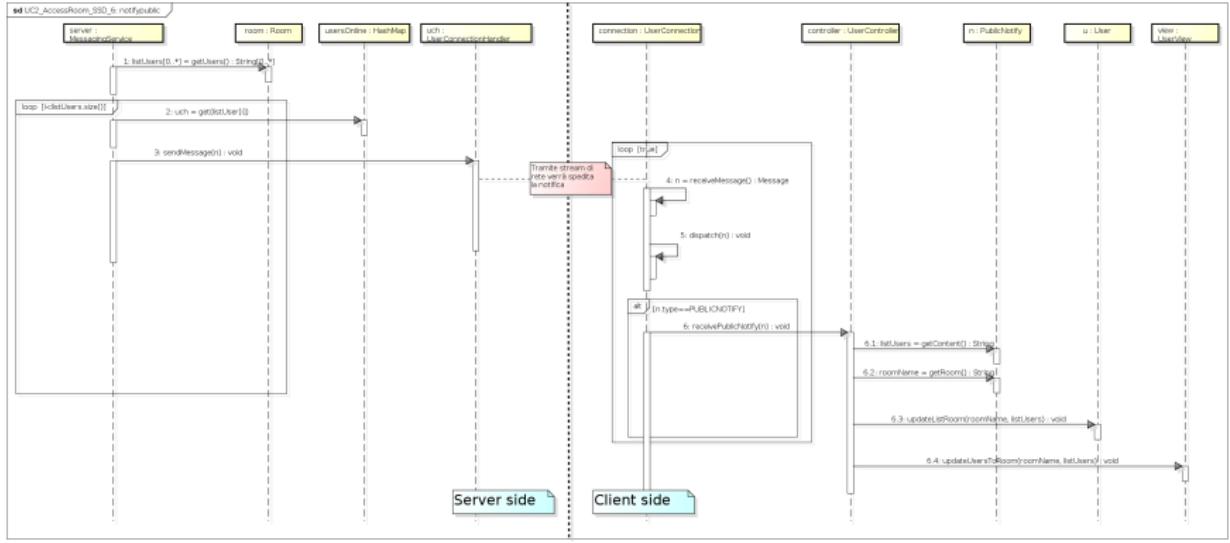
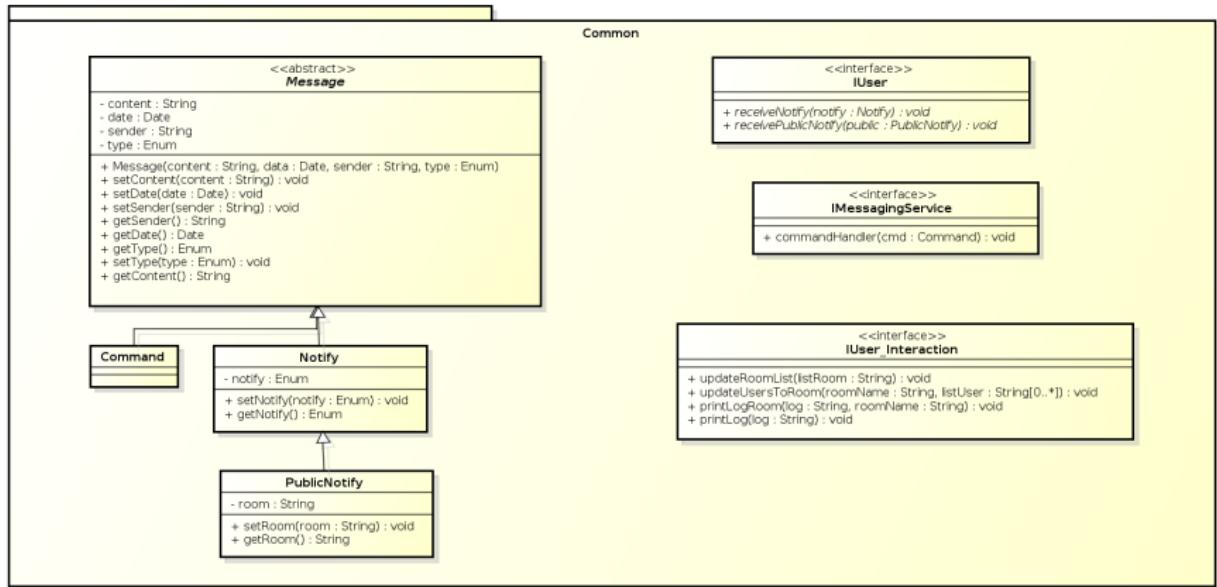


Figura 17 : SSD - OP6: `notifypublic(updateList)` del modello di dominio (figura 10)

# Iterazione 1 Progettazione: UC1 e UC2 Class Diagram - Common



**Figura 18 :** DCD - Diagramma delle Classi: Package Common

# Iterazione 1 Progettazione: UC1 e UC2 Class Diagram - Snuc Server

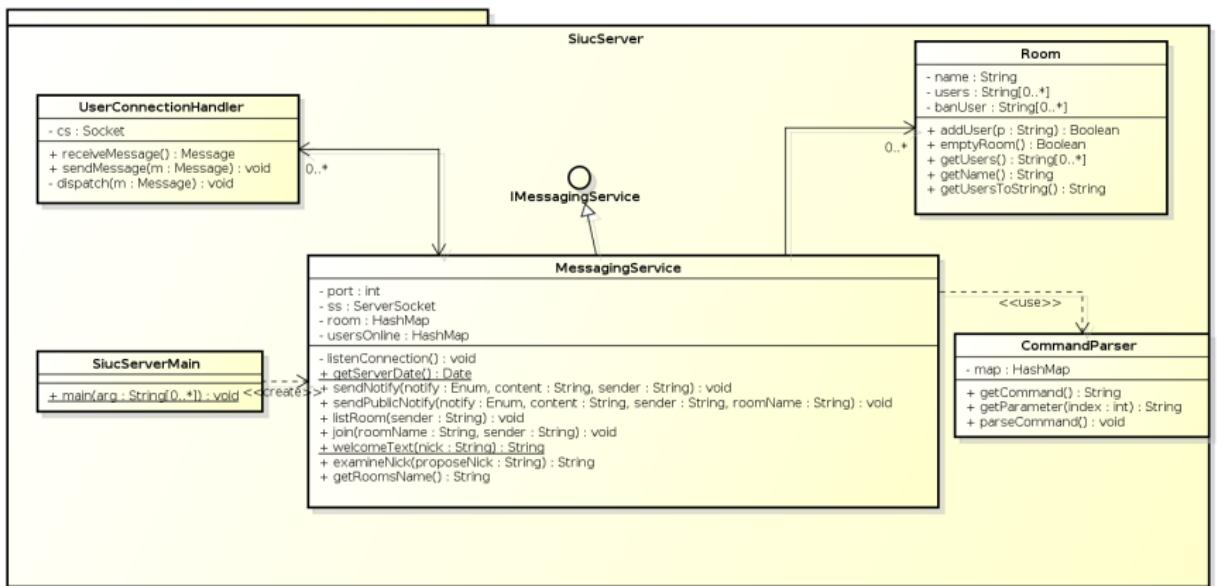


Figura 19 : DCD - Diagramma delle Classi: Package Snuc Server

## Iterazione 1 Progettazione: UC1 e UC2 Class Diagram - Snuc

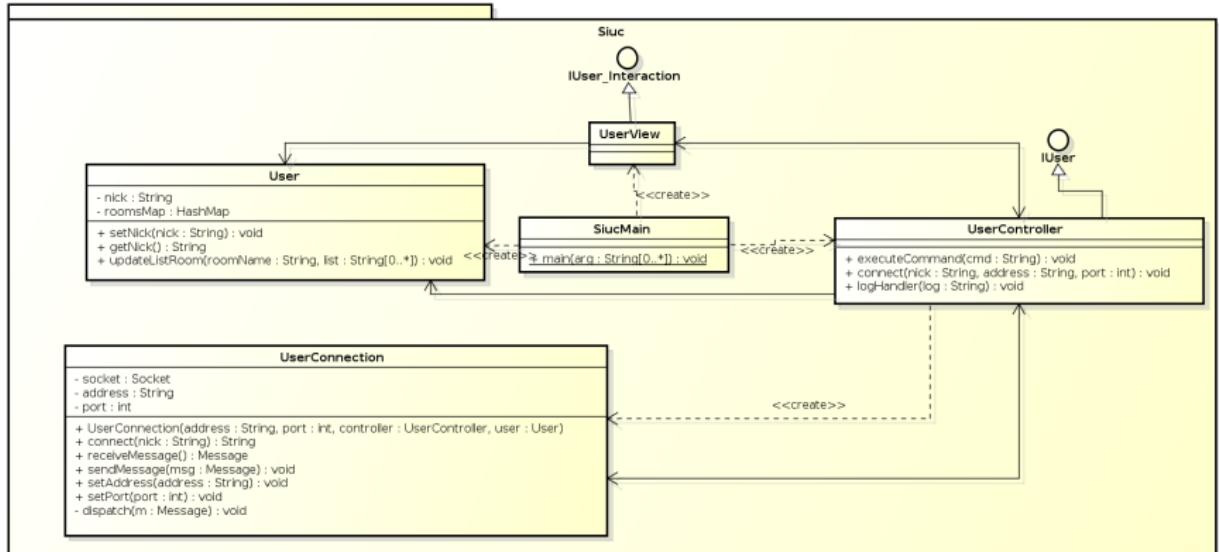


Figura 20 : DCD - Diagramma delle Classi: Package Snuc



## Iterazione 1: Implementazione - UC1 e UC2

*INSERIRE DESCRIZIONE*





## Iterazione 1: Test - UC1 e UC2

*INSERIRE DESCRIZIONE*





## Descrizione: Refactoring Iterazione 1

*INSERIRE DESCRIZIONE*



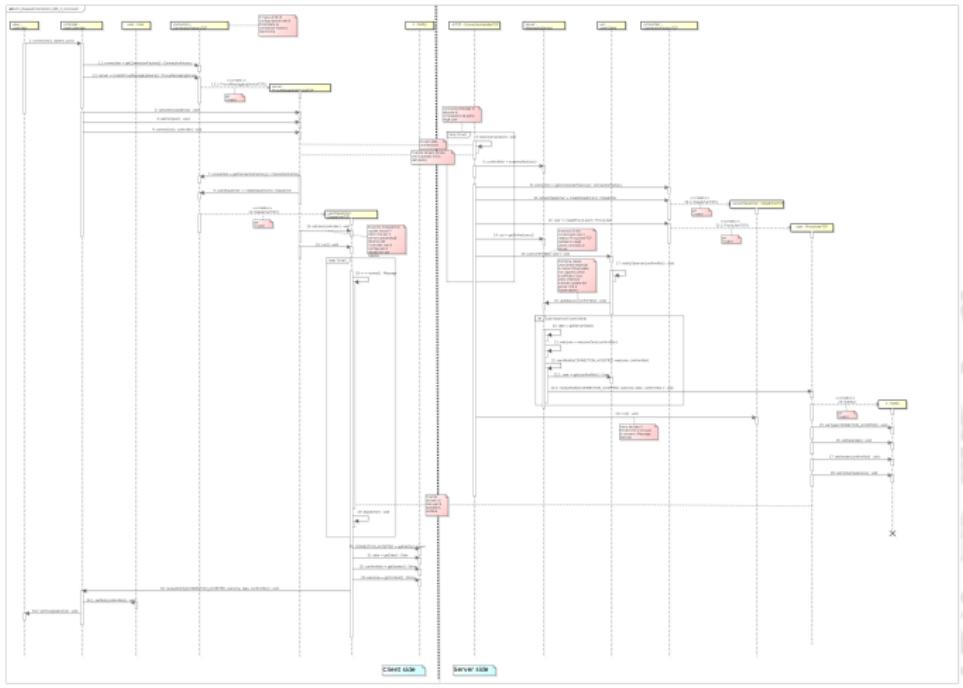


## Descrizione Refactoring Iterazione 1 - UC1\_RequestConnection

*INSERIRE DESCRIZIONE*



## Refactoring Iterazione 1: UC1\_RequestConnection - OP3-4





## Descrizione Refactoring Iterazione 1 - UC2\_AccessRoom

*INSERIRE DESCRIZIONE*



## Refactoring Iterazione 1: UC2\_AccessRoom - OP1

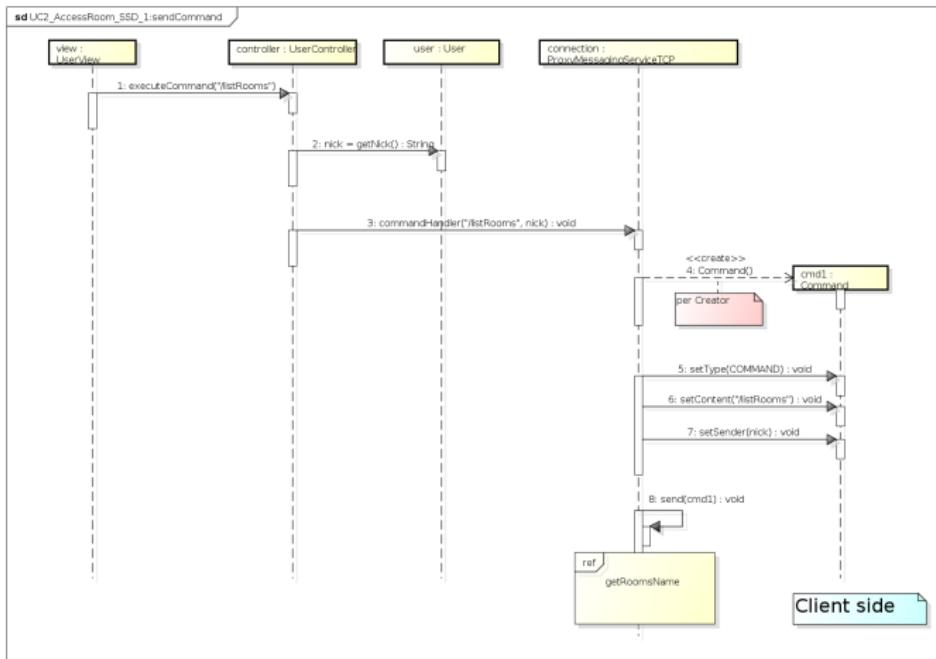
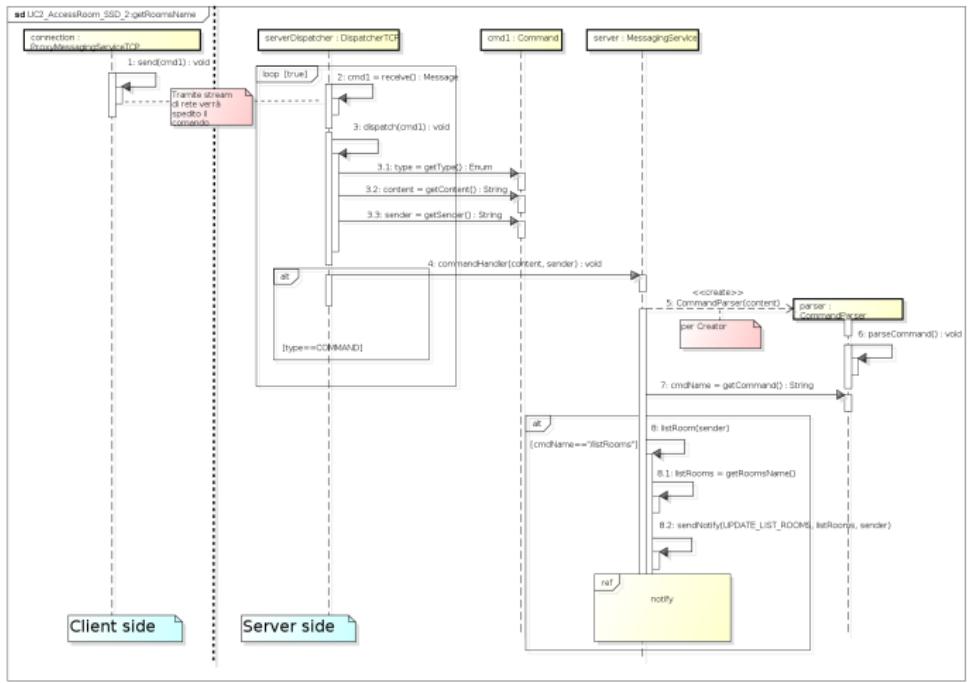


Figura 22 : SSD - OP1: sendCommand(cmd1) del modello di dominio (figura 10)

## Refactoring Iterazione 1: UC2\_AccessRoom - OP2



## Refactoring Iterazione 1: UC2\_AccessRoom - OP3

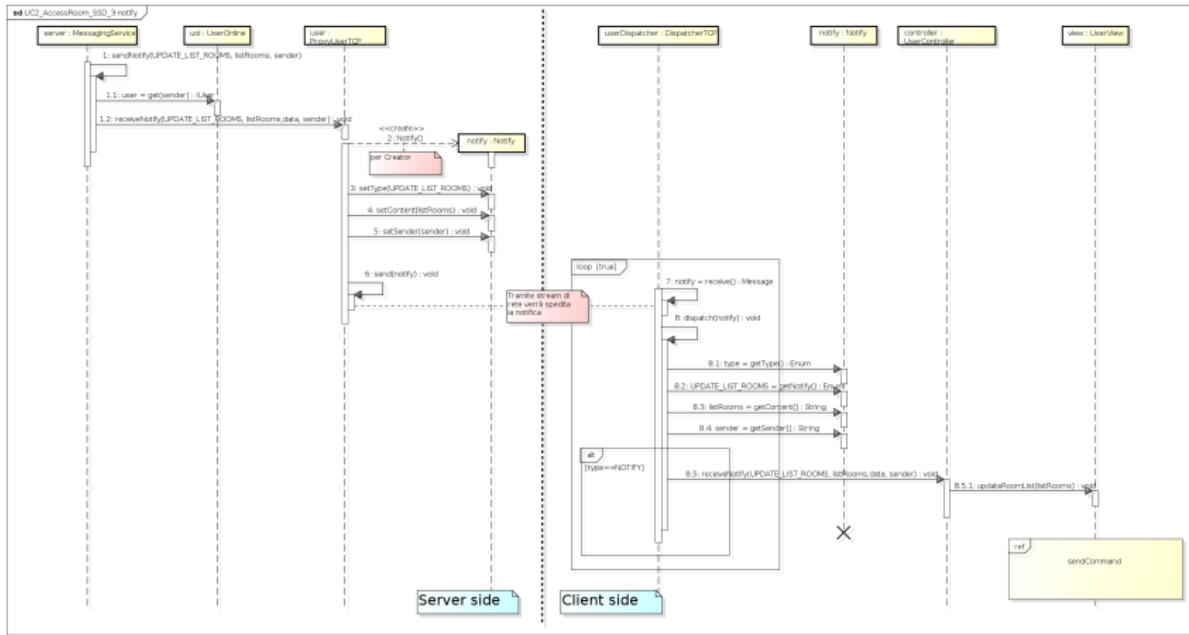
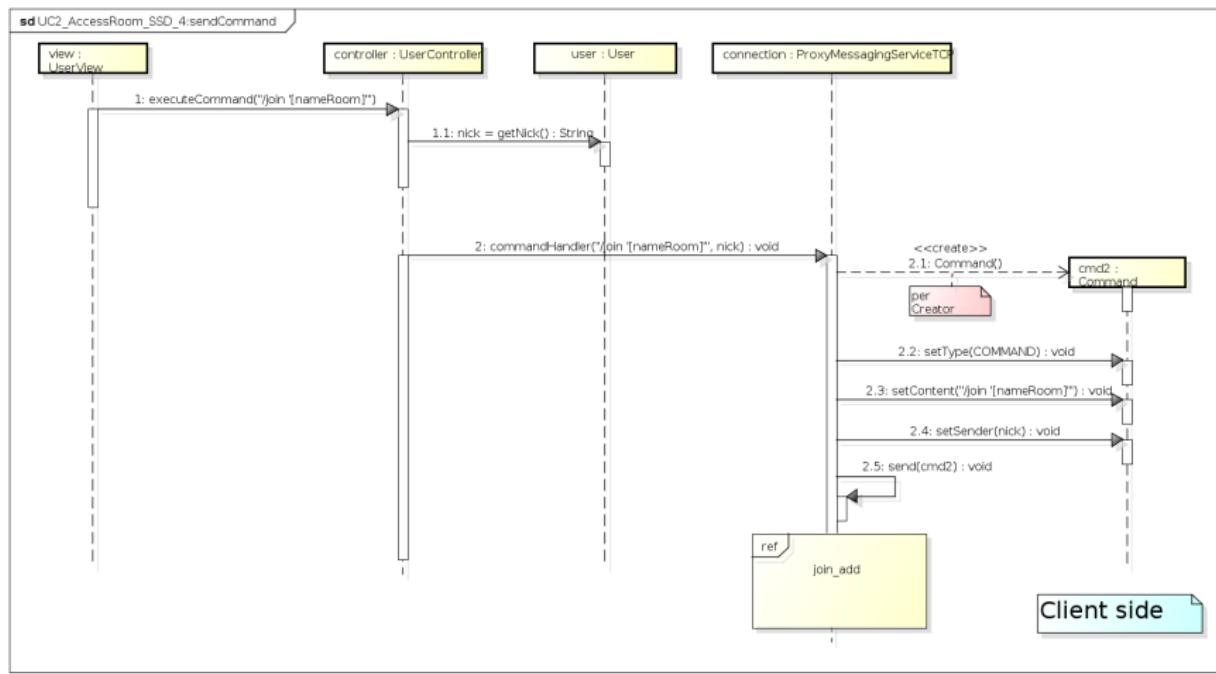
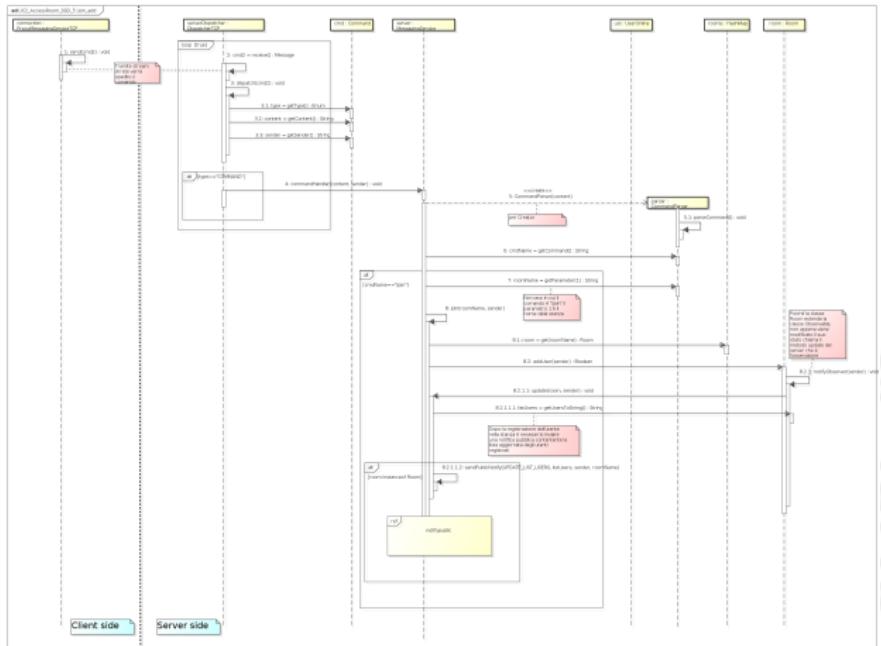


Figura 24 : SSD - OP3: `notify(list)` del modello di dominio (figura 10)

## Refactoring Iterazione 1: UC2\_AccessRoom - OP4

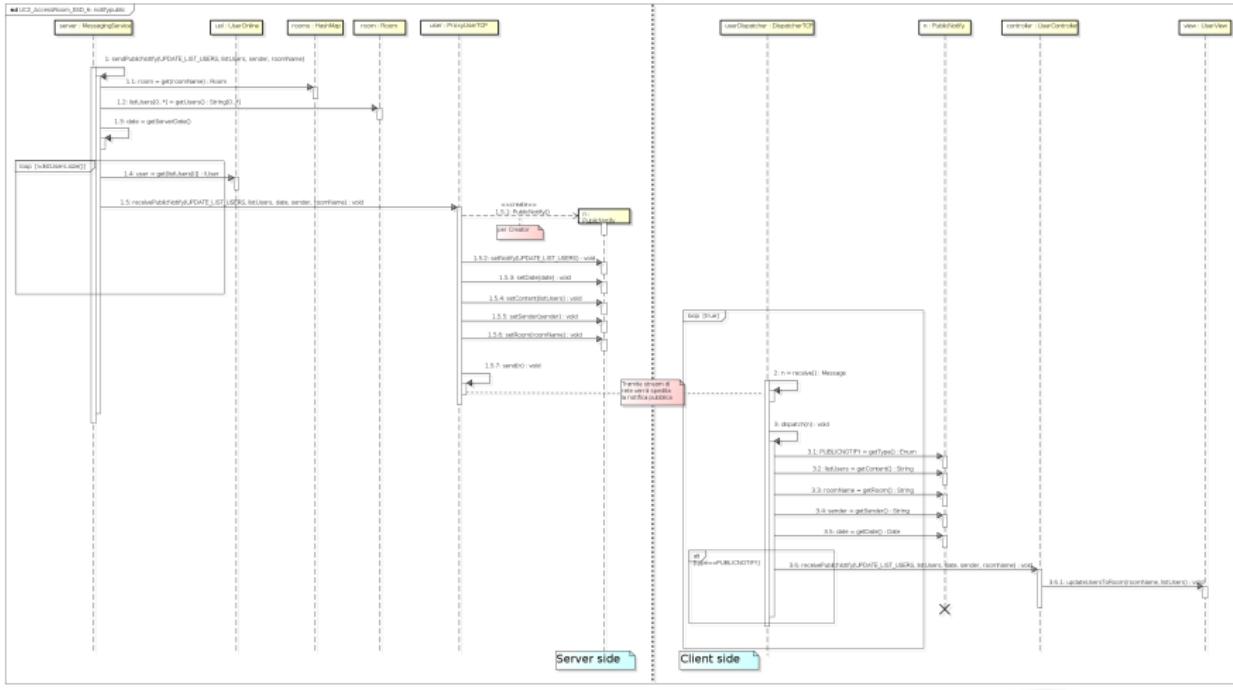


## Refactoring Iterazione 1: UC2\_AccessRoom - OP5



**Figura 26 : SSD - OP5: joinRoom(sender,room), addUserToRoom() del modello di dominio (figura 10)**

# Refactoring Iterazione 1: UC2\_AccessRoom - OP6



## Refactoring Iterazione 1: UC1 e UC2 Class Diagram - Common

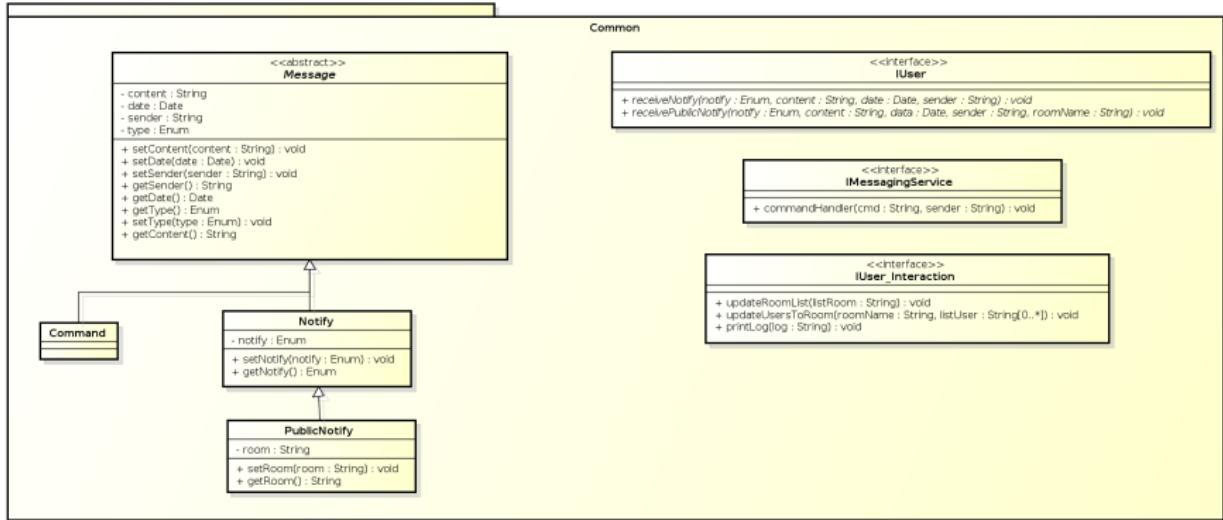


Figura 28 : DCD - Diagramma delle Classi: Package Common

## Refactoring Iterazione 1: UC1 e UC2 Class Diagram - Snuc

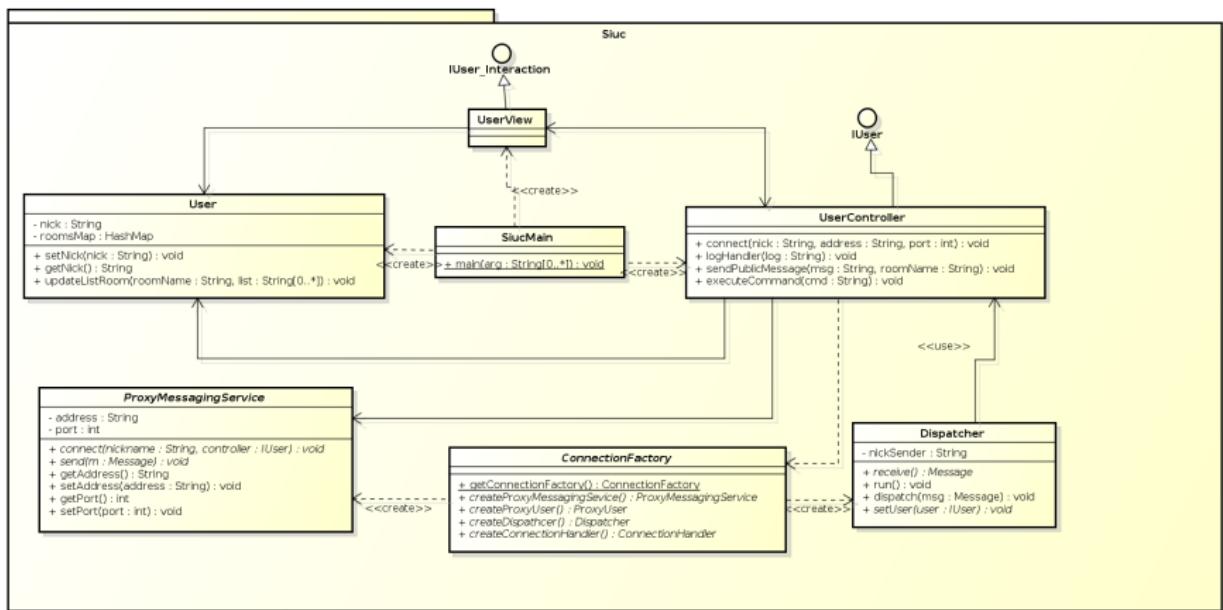


Figura 29 : DCD - Diagramma delle Classi: Package Snuc

## Refactoring Iterazione 1: UC1 e UC2 Class Diagram - Snuc Server

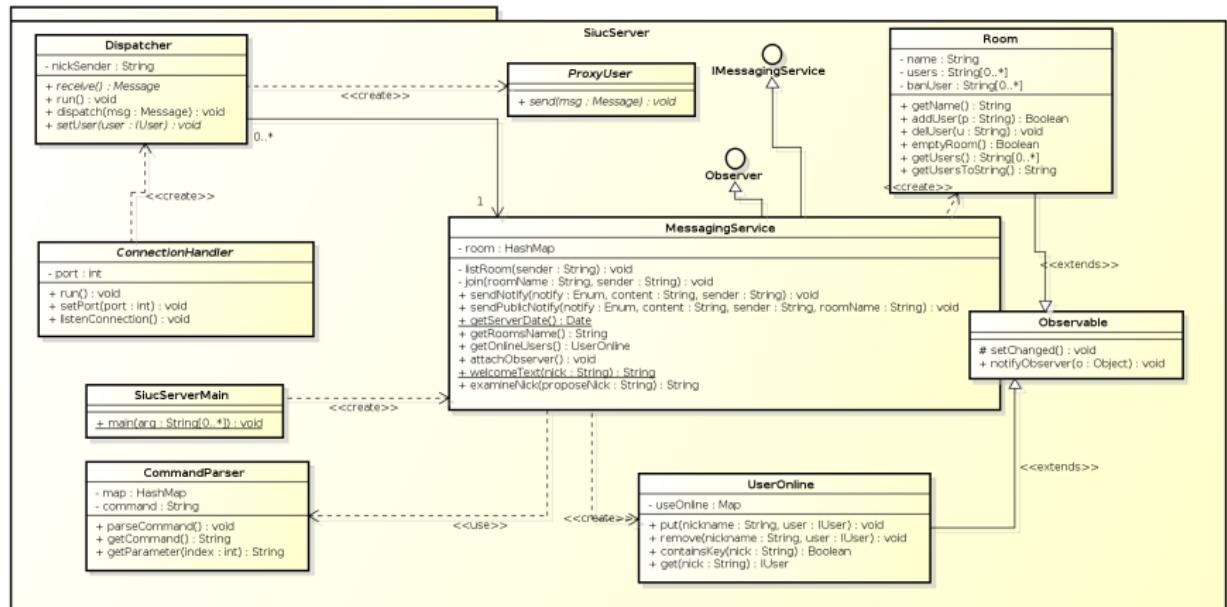


Figura 30 : DCD - Diagramma delle Classi: Package Snuc Server



# Refactoring Iterazione 1: Implementazione - UC1 e UC2

*INSERIRE DESCRIZIONE*





## Refactoring Iterazione 1: Test - UC1 e UC2

*INSERIRE DESCRIZIONE*





## Descrizione Elaborazione - Iterazione 2

*INSERIRE DESCRIZIONE*





## Iterazione 2: Requisiti - UC3\_SendPublicMessage

*INSERIRE DESCRIZIONE*





## Iterazione 2: Requisiti - UC3\_SendPublicMessage

**Tabella 5 :** Caso d'uso UC3\_SendPublicMessage

Nome caso d'uso	UC3_SendPublicMessage
Portata	Applicazione Smart Intelligent University Communications
Livello	Obiettivo Utente
Attore primario	SnucUser
Parti interessate e interessi	<ul style="list-style-type: none"> <li>▶ SnucUser, vuole che i messaggi siano inviati a ogni utente della stanza virtuale.</li> <li>▶ SnucAdmin, è interessato a supervisionare gli utenti del servizio affinché non ci siano abusi.</li> </ul>
Pre-condizioni	L'utente è registrato nella stanza in cui desidera inviare i messaggi.
Post-condizioni (garanzia di successo)	Ogni utente riceve il messaggio inviato.
Scenario principale di successo	<ol style="list-style-type: none"> <li>① L'utente inserisce in una opportuna area il messaggio da inviare.</li> <li>② Il messaggio viene inoltrato agli utenti presenti nella stanza selezionata.</li> </ol>



## Iterazione 2: Requisiti - UC3\_SendPublicMessage

Requisiti speciali (Requisiti Non Funzionali)	Comunicazione asincrona in cui lo scambio di informazioni avviene in tempo reale, senza sensibili pause tra invio e ricezione del messaggio
Elenco delle varianti tecnologiche	<ul style="list-style-type: none"> <li>▶ È possibile inviare messaggi confidenziali, autenticati e integri al server del servizio di messaggistica.</li> <li>▶ L'applicazione dovrebbe essere flessibile al funzionamento di diversi protocolli di comunicazione (es. TCP, UDP) e con diversi strati middleware (es. Socket, RMI)</li> </ul>
Frequenza di ripetizione	Potrebbe essere quasi ininterrotta
Varie e/o Problemi Aperti	//

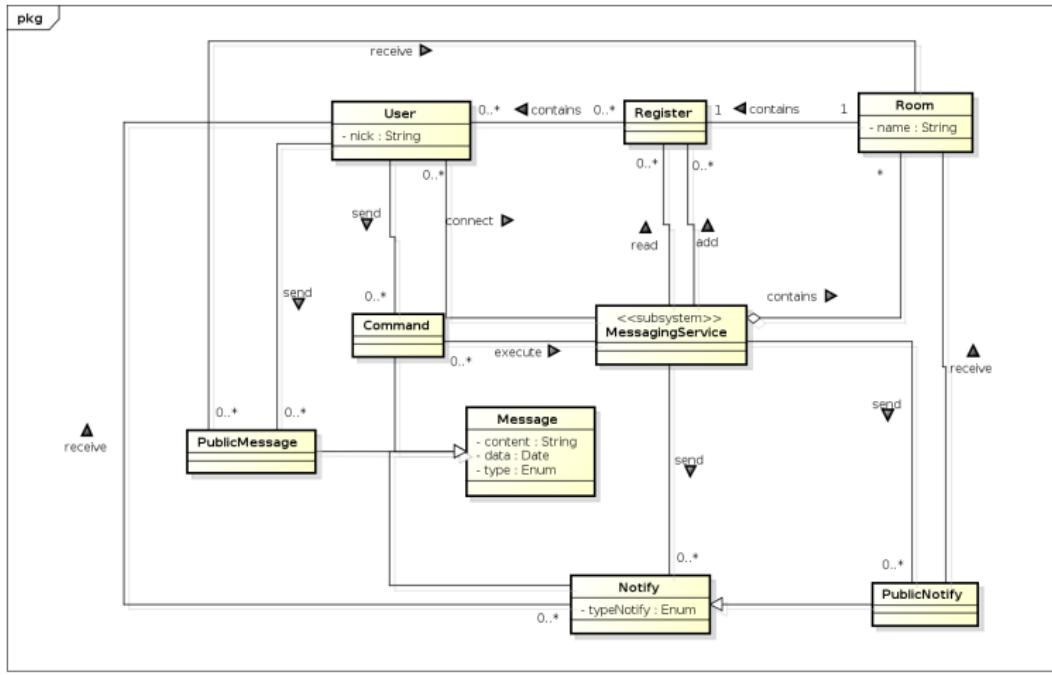


## Descrizione Iterazione 2: Analisi - UC3\_SendPublicMessage

*INSERIRE DESCRIZIONE*



## Iterazione 2: Analisi - UC3\_SendPublicMessage



## Iterazione 2: Analisi - UC3\_SendPublicMessage

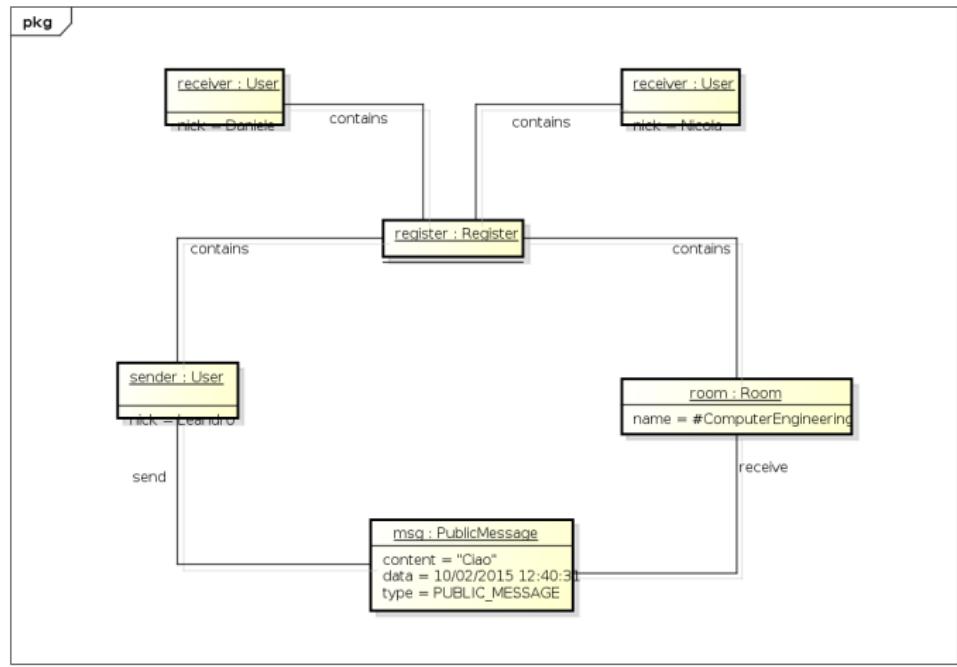
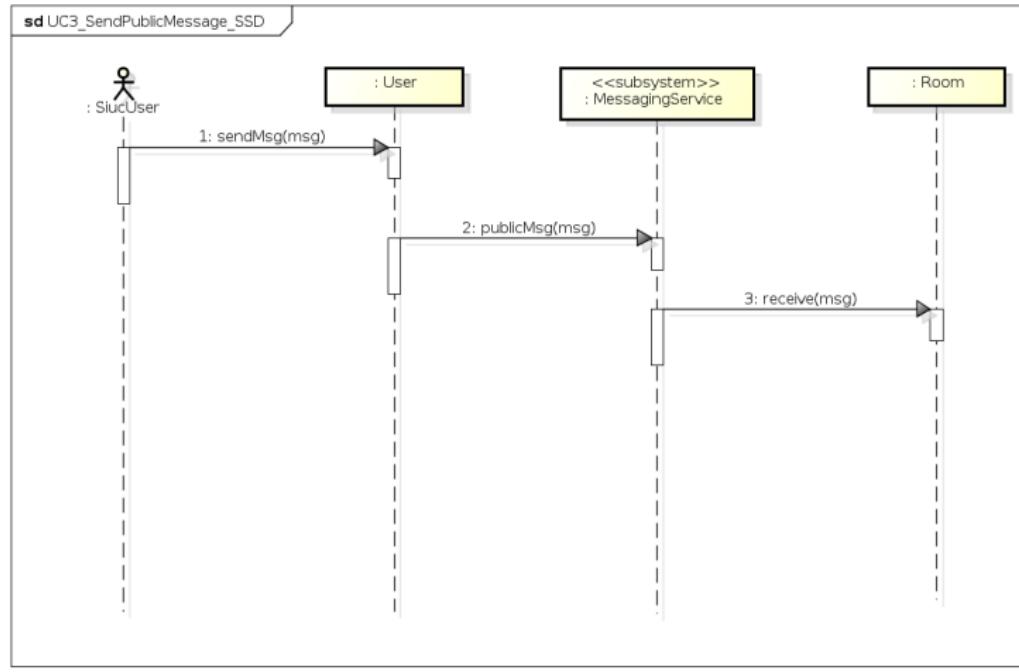


Figura 22 - UC3 - Oasi di informazione



## Iterazione 2: Analisi - UC3\_SendPublicMessage





## Iterazione 2: Analisi, UC2 contratto CO2 - NameCO2

**Tabella 6 :** UC2 Contratto CO2 - NameCO2

Operazione	
Riferimenti	
Pre-condizione	
Post-condizione	



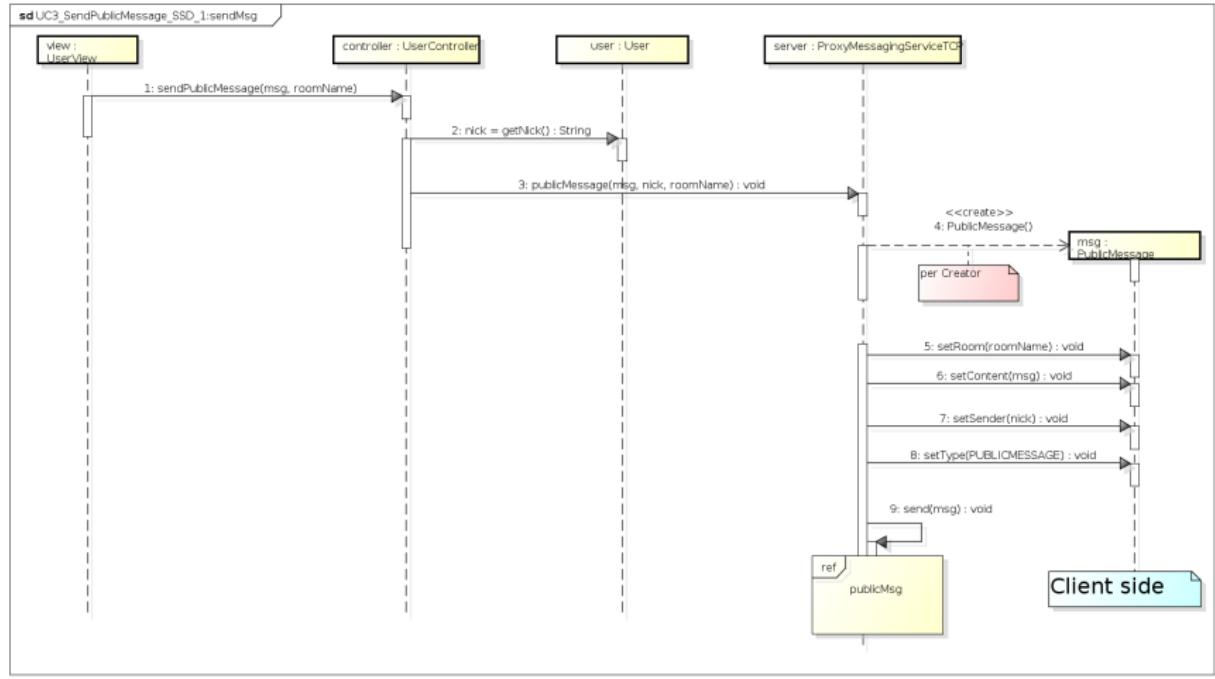


## Iterazione 2: Progettazione - UC3\_SendPublicMessage

*INSERIRE DESCRIZIONE*



## Iterazione 2 Progettazione: UC3\_SendPublicMessage - OP1



## Iterazione 2 Progettazione: UC3\_SendPublicMessage - OP2

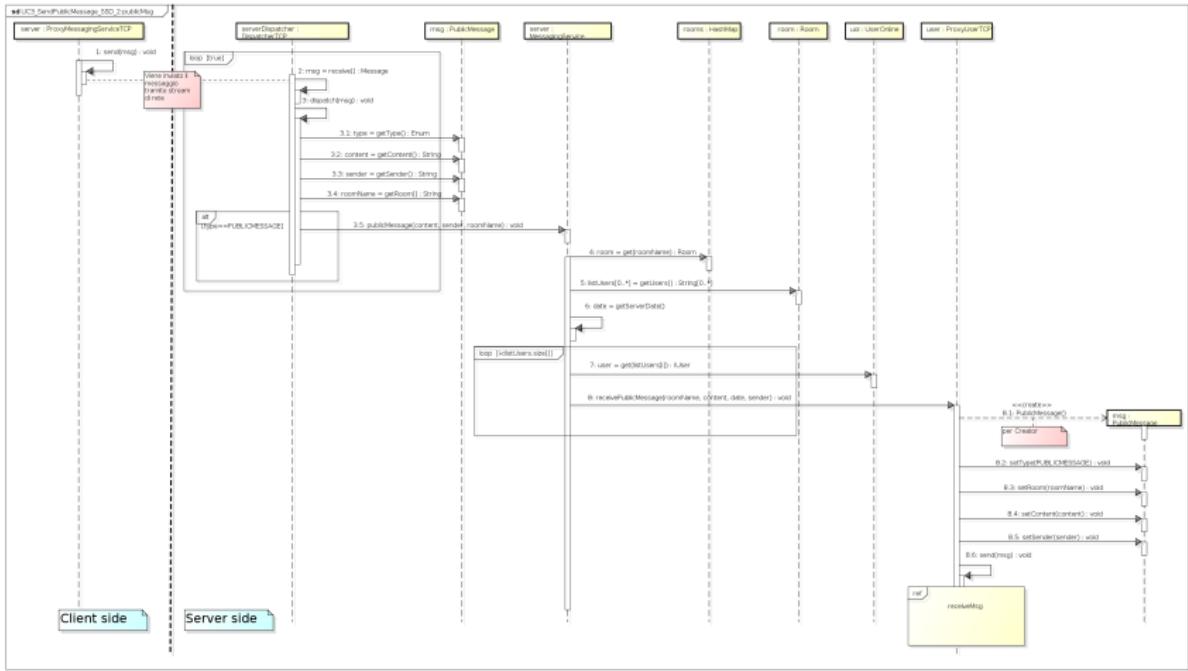


Figura 35 : SSD - OP2: pubblicMsg(msg) del modello di dominio (figura 33)

## Iterazione 2 Progettazione: UC3\_SendPublicMessage - OP3

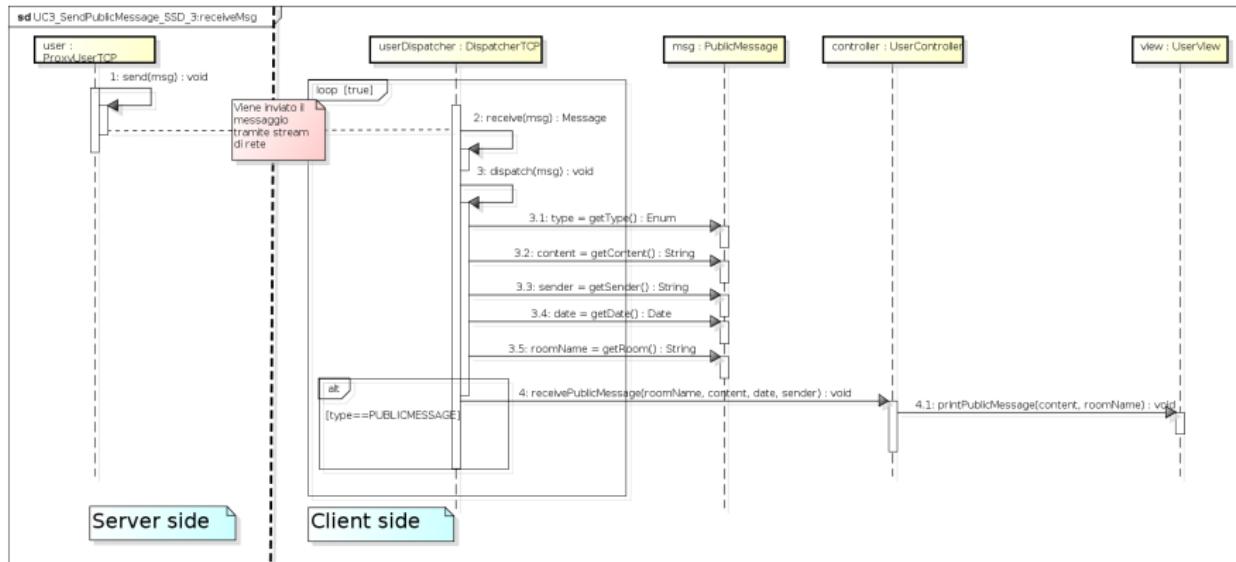


Figura 36 : SSD - OP3: receive(msg) del modello di dominio (figura 33)



## Iterazione 2 Progettazione: UC3 Class Diagram - Common

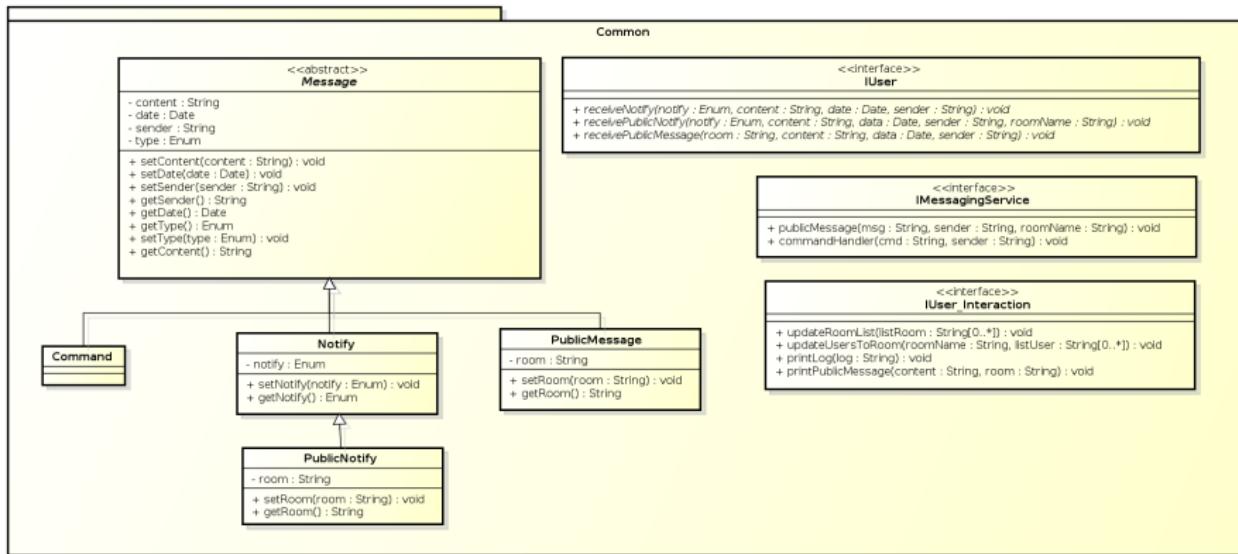


Figura 37 : DCD - Diagramma delle Classi: Package Common

## Iterazione 1 Progettazione: UC3 Class Diagram - Snuc

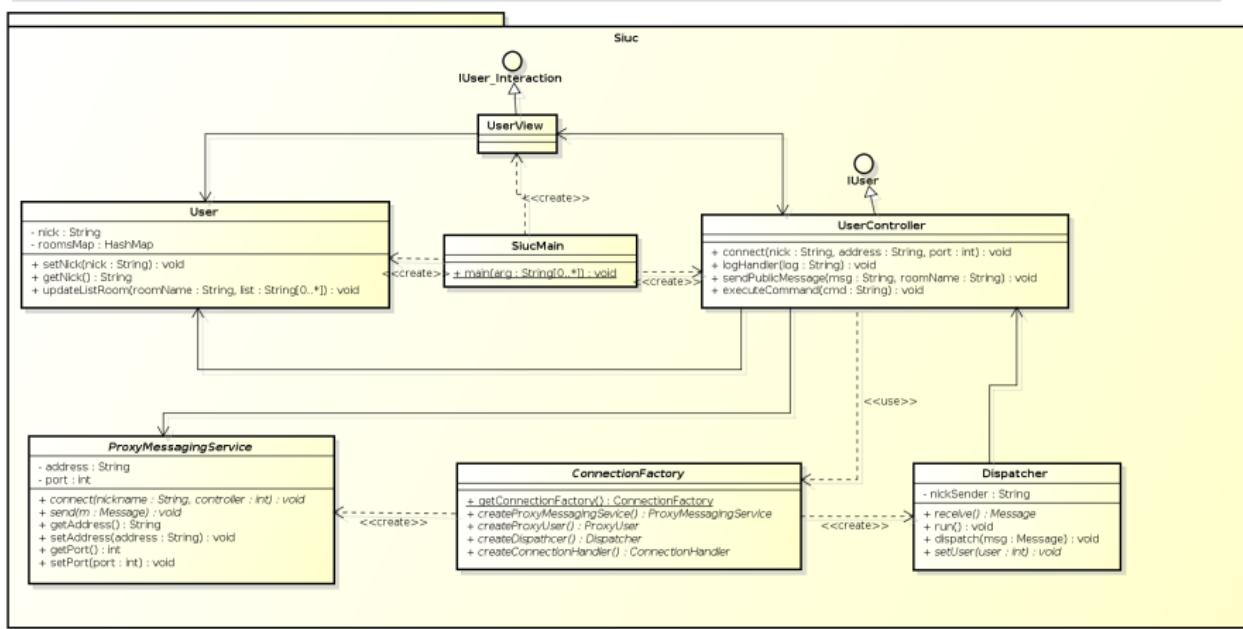
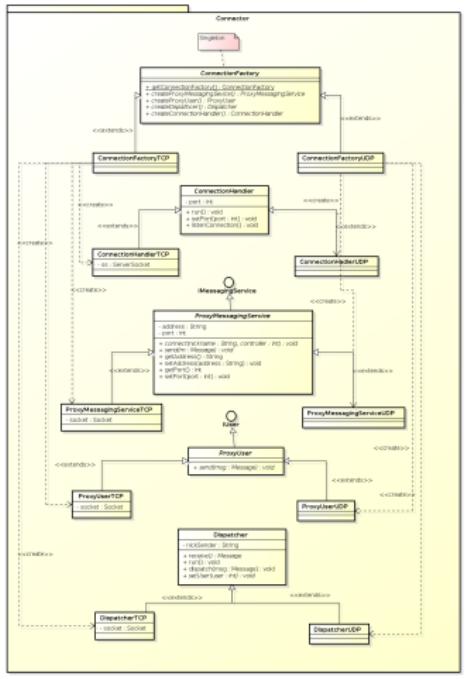


Figura 38 : DCD - Diagramma delle Classi: Package Snuc

## Iterazione 2 Progettazione: UC3 Class Diagram - Connector



## Iterazione 2 Progettazione: UC3 Class Diagram - Snuc Server

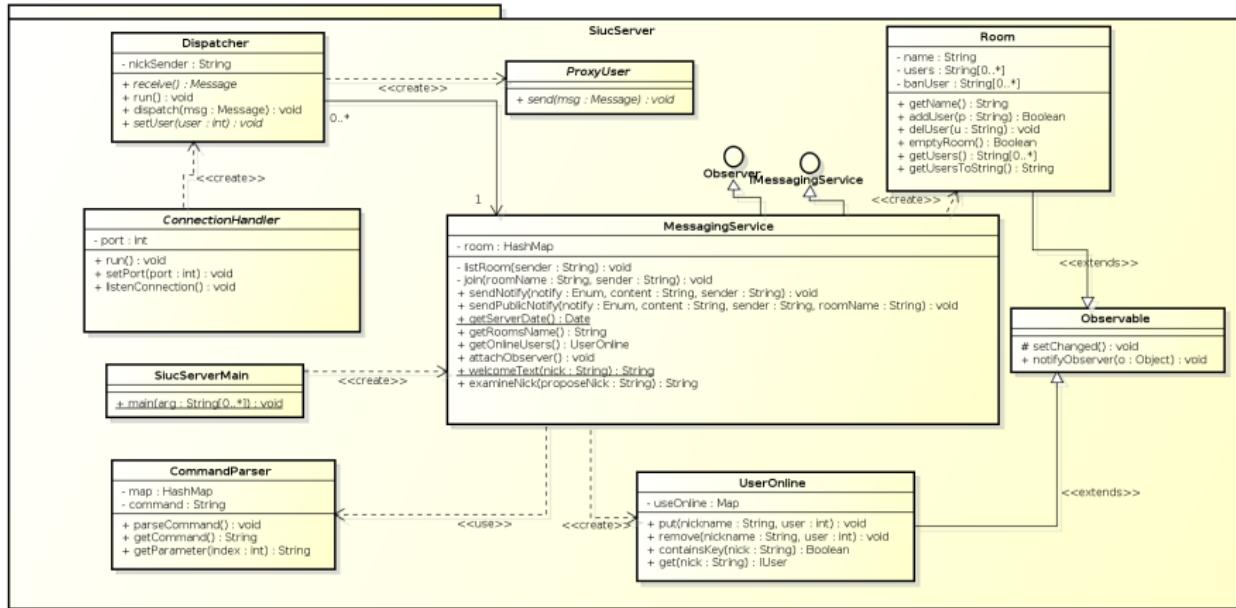


Figura 40 : DCD - Diagramma delle Classi: Package Snuc Server



## Iterazione 2: Implementazione - UC3\_SendPublicMessage

*INSERIRE DESCRIZIONE*





## Iterazione 2: Test - UC3\_SendPublicMessage

*INSERIRE DESCRIZIONE*





## Descrizione Elaborazione: Iterazione 3

*INSERIRE DESCRIZIONE*





## Iterazione 3: Requisiti - UC4\_SendPrivateMessage

**Tabella 7 :** Caso d'uso UC4\_SendPrivateMessage

Nome caso d'uso	UC3_SendRoomMessage
Portata	Applicazione Smart Intelligent University Communications
Livello	Obiettivo Utente
Attore primario	SnucUser
Parti interessate e interessi	SnucUser: vuole che i messaggi siano inviati all'utente selezionato presente nella stanza virtuale
Pre-condizioni	L'utente è registrato nella stanza in cui desidera inviare un messaggio ad un altro utente presente
Post-condizioni (garanzia di successo)	L'utente selezionato riceve il messaggio inviato
Scenario principale di successo	<ul style="list-style-type: none"><li>① L'utente seleziona il destinatario del messaggio privato.</li><li>② L'utente inserisce da tastiera il messaggio da inviare.</li><li>③ Il messaggio viene inoltrato al destinatario selezionato.</li></ul>



## Iterazione 3: Requisiti - UC4\_SendPrivateMessage

Requisiti speciali (Requisiti Non Funzionali)	Comunicazione asincrona in cui lo scambio di informazioni avviene in tempo reale, senza sensibili pause tra invio e ricezione del messaggio
Elenco delle varianti tecnologiche	<ul style="list-style-type: none"> <li>▶ È possibile inviare messaggi confidenziali, autenticati e integri al server del servizio di messaggistica.</li> <li>▶ L'applicazione dovrebbe essere flessibile al funzionamento di diversi protocolli di comunicazione (es. TCP, UDP) e con diversi strati middleware (es. Socket, RMI)</li> </ul>
Frequenza di ripetizione	Potrebbe essere quasi ininterrotta
Varie e/o Problemi Aperti	//

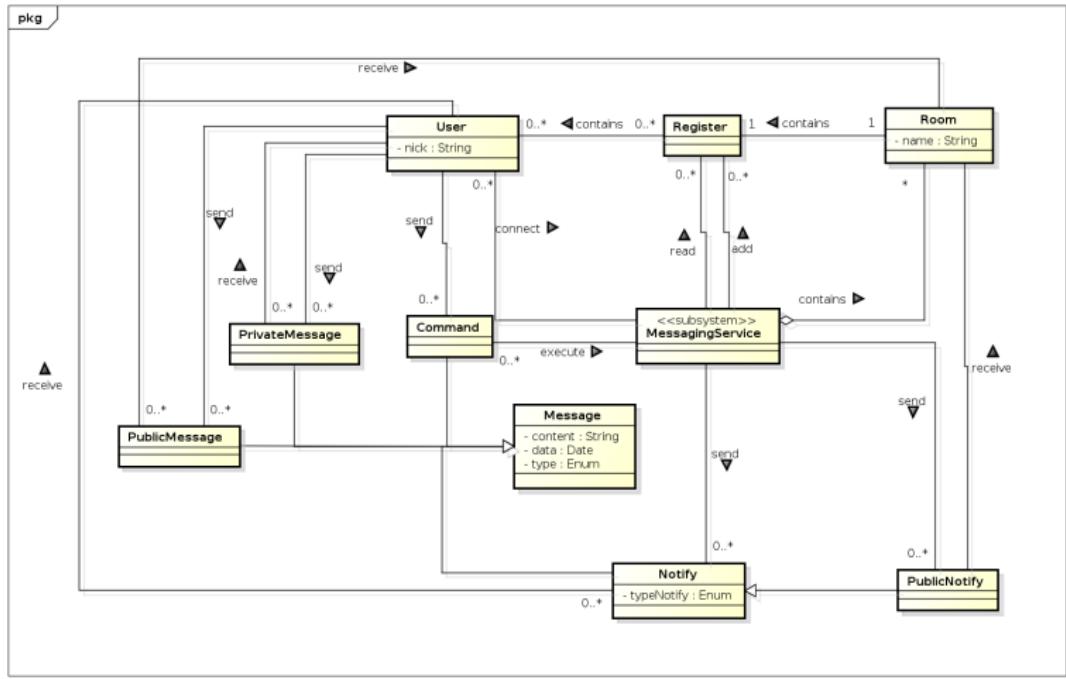


## Descrizione Iterazione 3: Analisi - UC4\_SendPrivateMessage

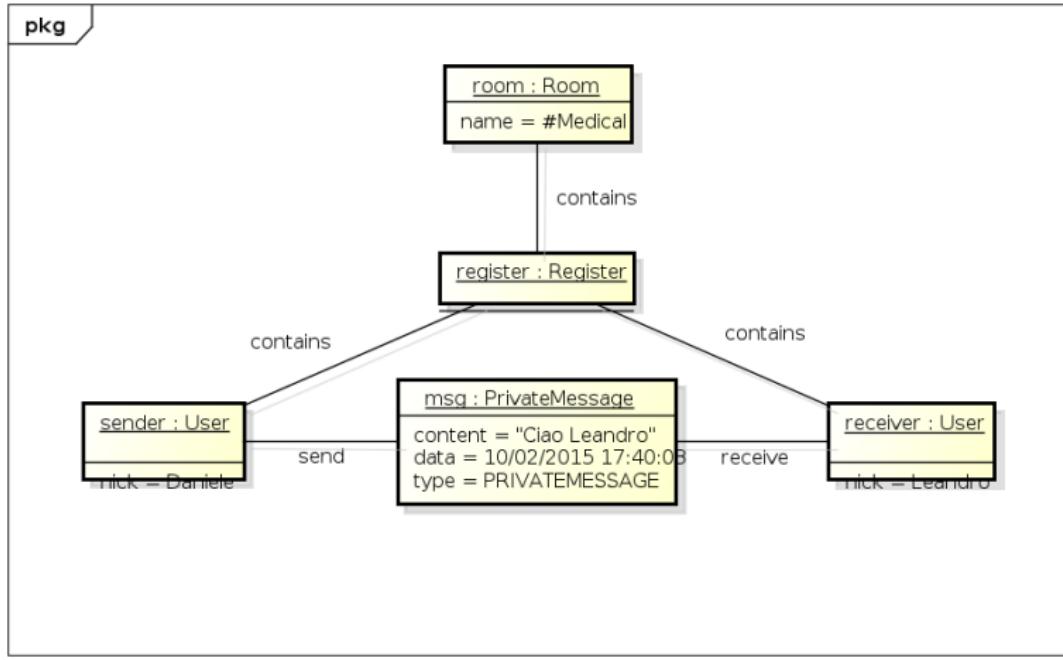
*INSERIRE DESCRIZIONE*



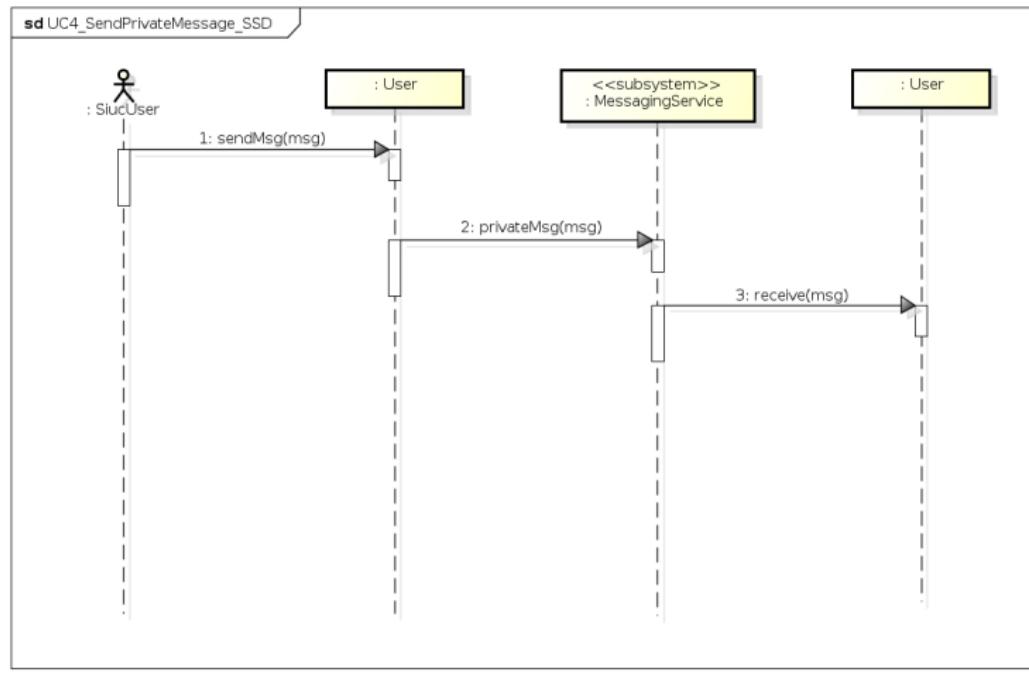
## Iterazione 3: Analisi - UC4\_SendPrivateMessage



## Iterazione 3: Analisi - UC4\_SendPrivateMessage



## Iterazione 3: Analisi - UC4\_SendPrivateMessage





## Iterazione 3: Analisi, UC3 contratto CO3 - NameCO3

**Tabella 8 :** UC3 Contratto CO3 - NameCO3

Operazione	
Riferimenti	
Pre-condizione	
Post-condizione	





## Iterazione 3: Progettazione - UC4\_SendPrivateMessage

*INSERIRE DESCRIZIONE*



## Iterazione 3 Progettazione: UC4\_SendPrivateMessage - OP1

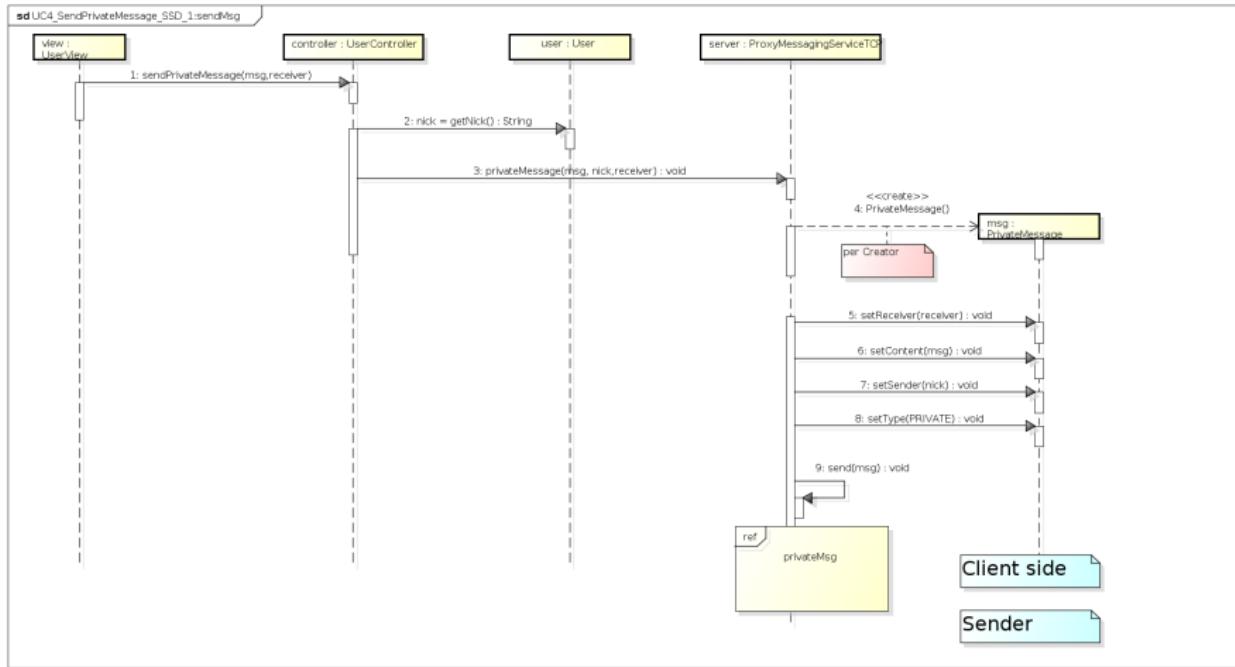
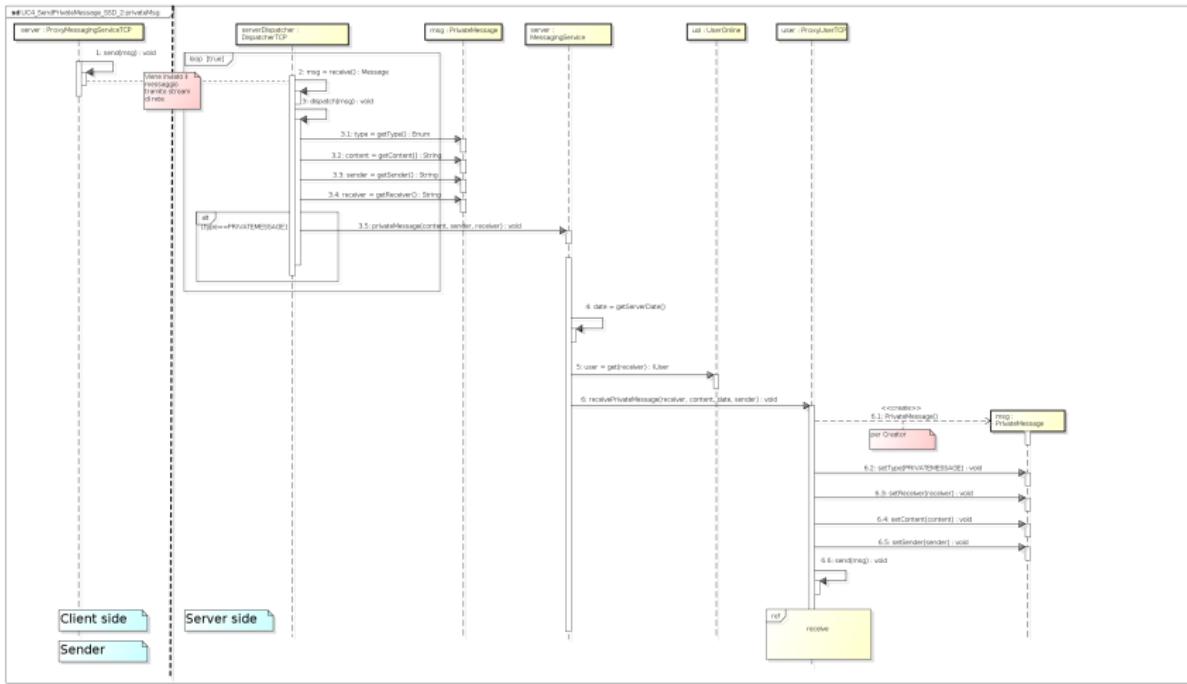


Figura 44 : SSD - OP1: sendMsg(msg) del modello di dominio (figura 42)

## Iterazione 3 Progettazione: UC4\_SendPrivateMessage - OP2



## Iterazione 3 Progettazione: UC4\_SendPrivateMessage - OP3

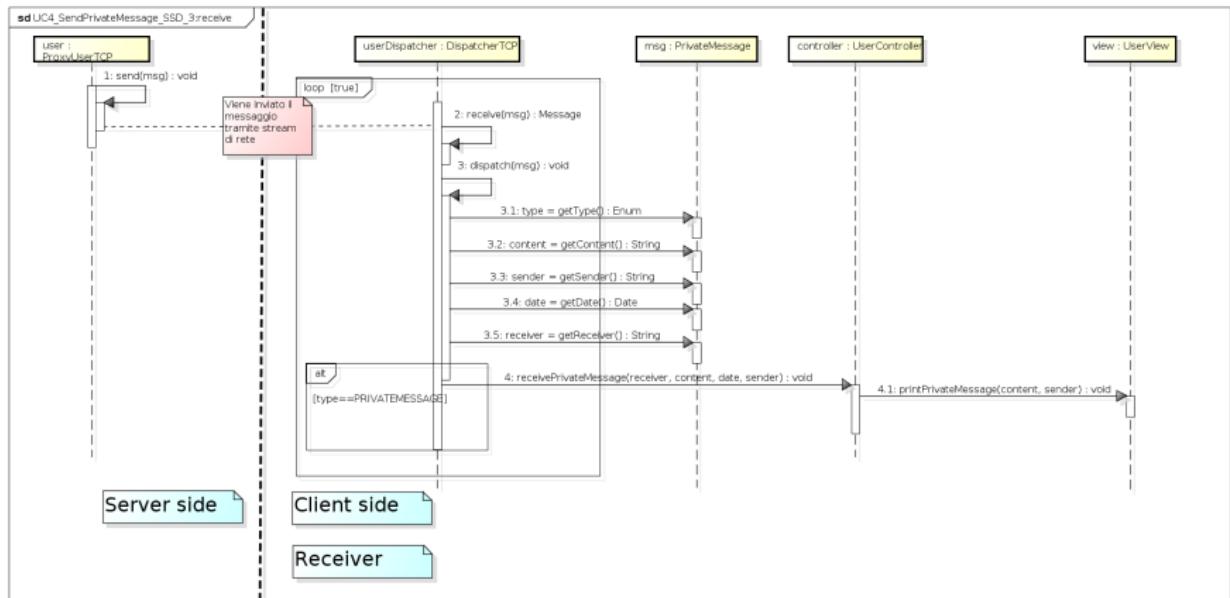


Figura 46 : SSD - OP3: receive(msg) del modello di dominio (figura 43)

## Iterazione 3 Progettazione: UC4 Class Diagram - Common

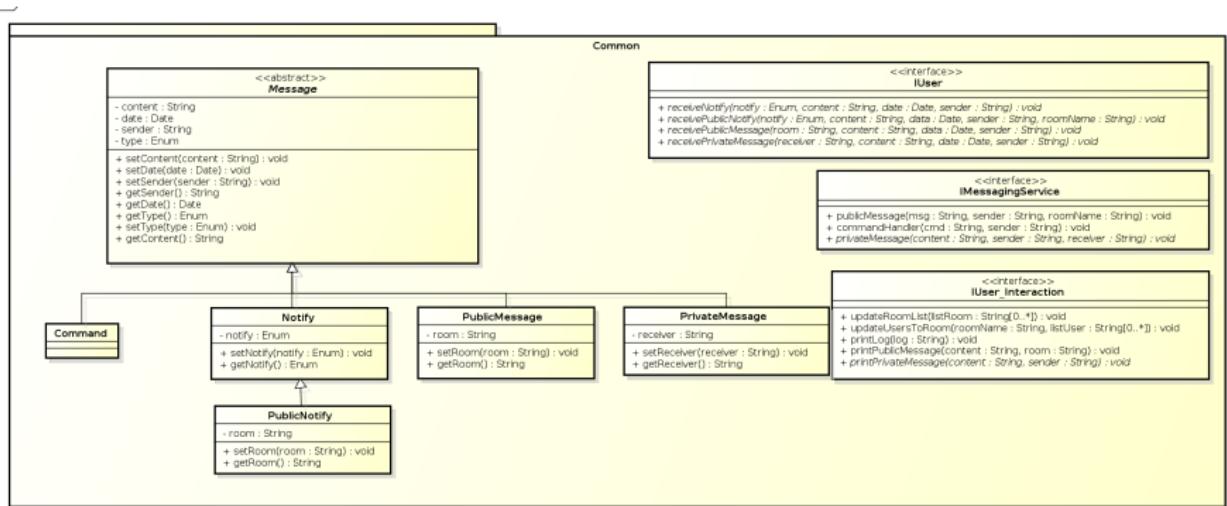


Figura 47 : DCD - Diagramma delle Classi: Package Common

## Iterazione 3 Progettazione: UC4 Class Diagram - Snuc

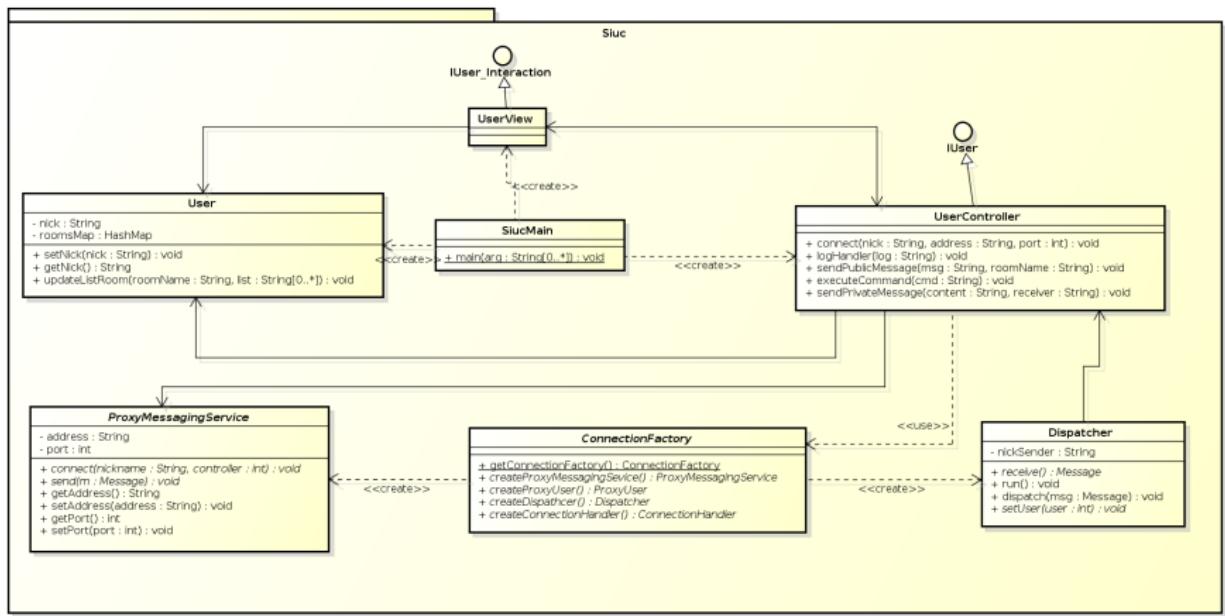
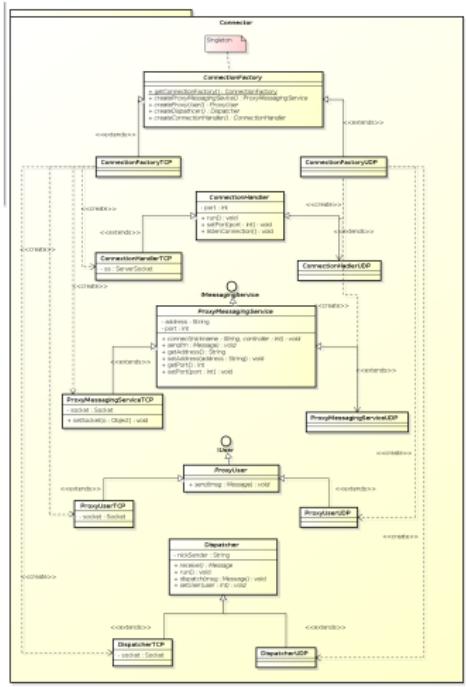


Figura 48 : DCD - Diagramma delle Classi: Package Snuc

## Iterazione 3 Progettazione: UC4 Class Diagram - Connector



## Iterazione 3 Progettazione: UC4 Class Diagram - Snuc Server

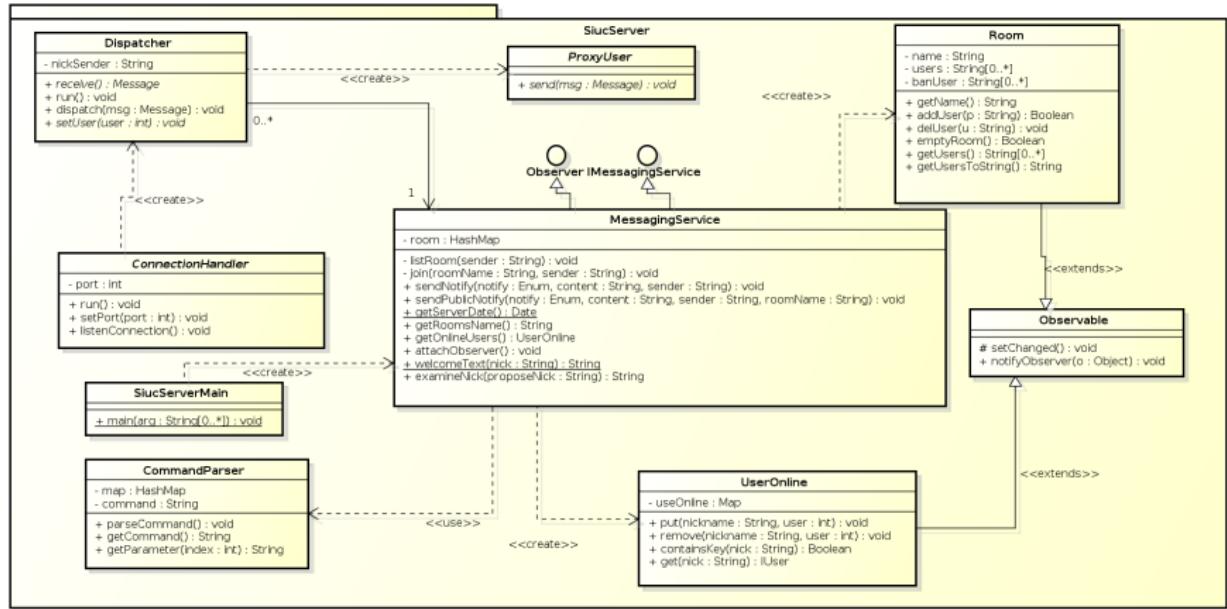


Figura 50 : DCD - Diagramma delle Classi: Package Snuc Server



## Iterazione 3: Implementazione - UC4\_SendPrivateMessage

*INSERIRE DESCRIZIONE*





## Iterazione 3: Test - UC4\_SendPrivateMessage

*INSERIRE DESCRIZIONE*





## Riferimenti bibliografici



**Herbert Schildt**

*Java SE 7. La guida completa.*  
McGraw-Hill, 2012



**Martin Fowler, L. Baresi, S. Gaburri**

*UML distilled. Guida rapida al linguaggio di modellazione standard*  
Pearson Education Italia, 2010



**Craig Larman, Luca Cabibbo**

*Applicare UML e i pattern: analisi e progettazione orientata agli oggetti*  
Pearson Education Italia, 2005



**M.L. Liu**

*Distributed Computing: Principles and Applications*  
Addison-Wesley, 2003



**Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides**

*Design patterns - Elementi per il riuso di software ad oggetti*  
Pearson Education Italia, 1995



**Pattern GOF - Giuseppe Dell'Abate's Blog**

Riferimenti: <https://dellabate.wordpress.com/category/gof-pattern/>



**Test unità con Junit4**

Riferimenti: <http://junit.org>





## Riferimenti bibliografici



### The Java Tutorials

Riferimenti: <http://docs.oracle.com/javase/tutorial/>



### Java Platform, Standard Edition 8 API Specification

Riferimenti: <http://docs.oracle.com/javase/8/docs/api/>



### Git –everything-is-local

Riferimenti: <http://git-scm.com/>



### git - la guida tascabile

Riferimenti: <http://rogerdudler.github.io/git-guide/index.it.html>



### Astah.net: UML and Modeling Tools

Riferimenti: <http://astah.net/>



### NetBeans Documentation

Riferimenti: <https://netbeans.org/kb/docs/java/quickstart.html>