PC Ready: Iterazione 3

Sommario

[Introduzione alla terza iterazione 1](#_Toc32314421)

[Modello di dominio 2](#_Toc32314422)

[Diagramma delle classi di progetto 3](#_Toc32314423)

[UC5 5](#_Toc32314424)

[UC6 8](#_Toc32314425)

[UC7 9](#_Toc32314426)

# Introduzione alla terza iterazione

Terminata la seconda iterazione si è ottenuto un programma dotato di console in grado di eseguire le operazioni di creazione delle configurazioni con tutti i dovuti controlli di compatibilità.

Nella suddetta iterazione, in virtù di quanto fatto in precedenza, si è pensato di introdurre il caso d’uso UC5 il quale consente di far effettuare ad un cliente un acquisto di una componente o di una configurazione, e di conseguenza salvare nella memoria del sistema un ordine e le informazioni relative al metodo di pagamento scelto dal cliente.

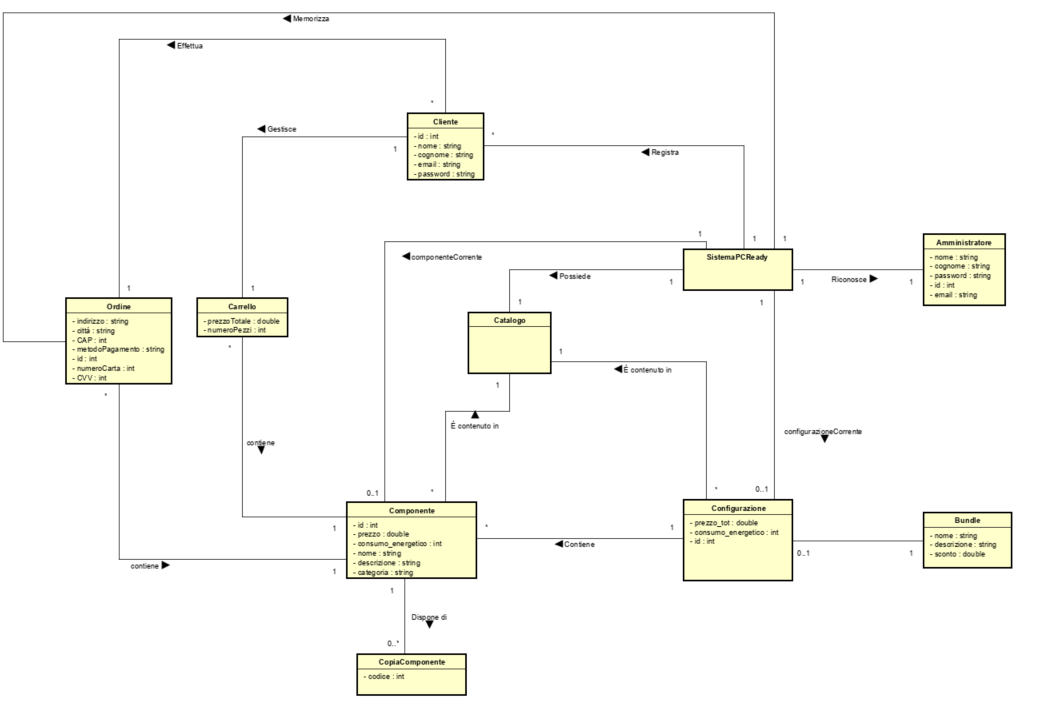
Un’ulteriore punto fondamentale di questa nuova iterazione è la registrazione di un cliente e il login di quest’ultimo o di un amministratore, descritti con le eventuali eccezioni dovute ad un inserimento errato delle credenziali.

L’introduzione del login consentirà di introdurre un “accesso alle operazioni” basato sul fatto che sia un cliente o un amministratore a richiedere il servizio: ad esempio si vuole che il cliente non sia in grado di accedere alle funzionalità di creazione di un bundle, la quale è riservata ad un amministratore

Ecco un breve elenco puntato dei contenuti previsti per tale iterazione:

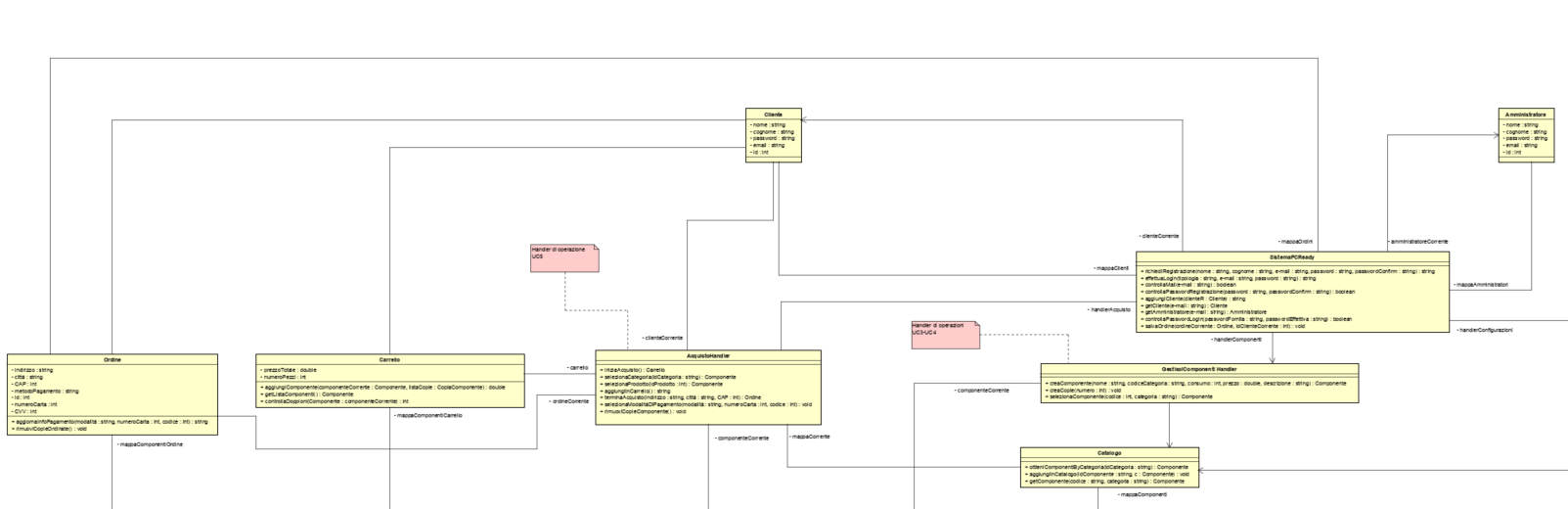
* Sviluppo dei casi d’uso UC5, UC6 e UC7 in forma completa
* Aggiornamento del modello di dominio e diagramma delle classi di progetto con le nuove classi introdotte dai casi d’uso sopra citati
* Sviluppo dei diagrammi di sequenza di sistema e dei diagrammi di sequenza per UC5,UC6,UC7
* Implementazione dei casi d’uso UC5, UC6, UC7 con le rispettive estensioni
* Migliorie dell’interfaccia utente e aggiunta dei nuovi comandi di login, registrazione e acquisto

# Modello di dominio

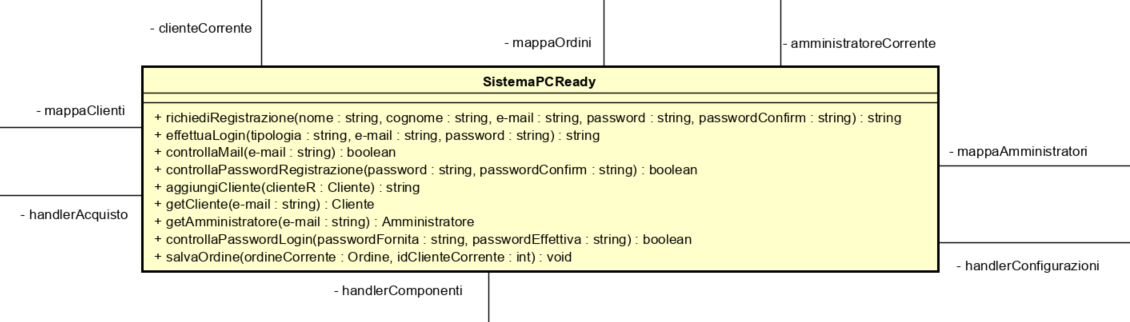


# Diagramma delle classi di progetto

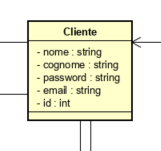
Ecco riportata la struttura del diagramma delle classi di progetto relativa ai casi d’uso sviluppati nel corso di questa iterazione: si riporta prima una panoramica del diagramma, e successivamente il dettaglio delle classi



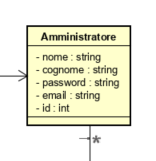
* **SistemaPCReady**



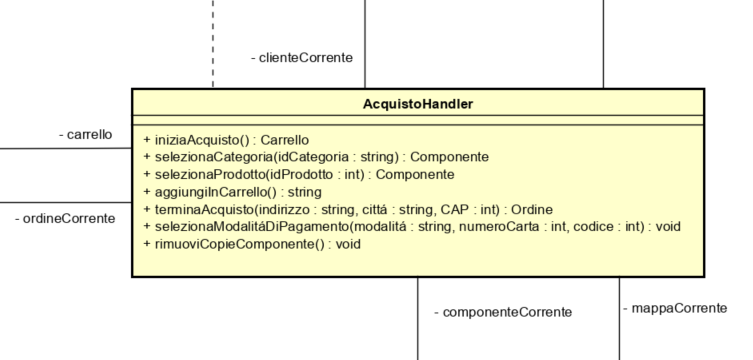
* **Cliente**



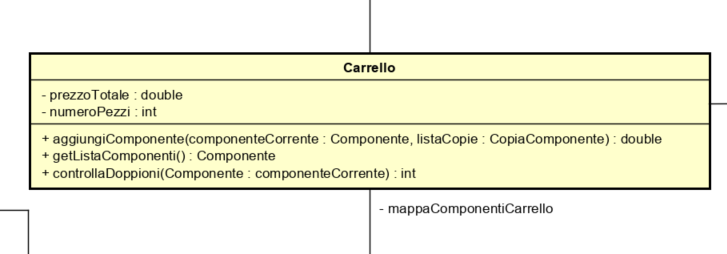
* **Amministratore**



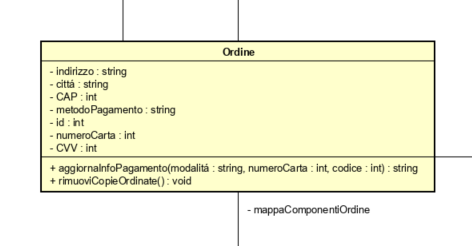
* **AcquistoHandler**



* **Carrello**



* **Ordine**



# UC5

Tale caso d’uso si compone di 6 operazioni fondamentali, le quali vengono gestite da un ulteriore handler di caso d’uso, mediante il pattern “Controller”, ai fini di ridurre le responsabilità proprie della classe SistemaPCReady e rispettare i principi di “Low Coupling” e “High Cohesion”.

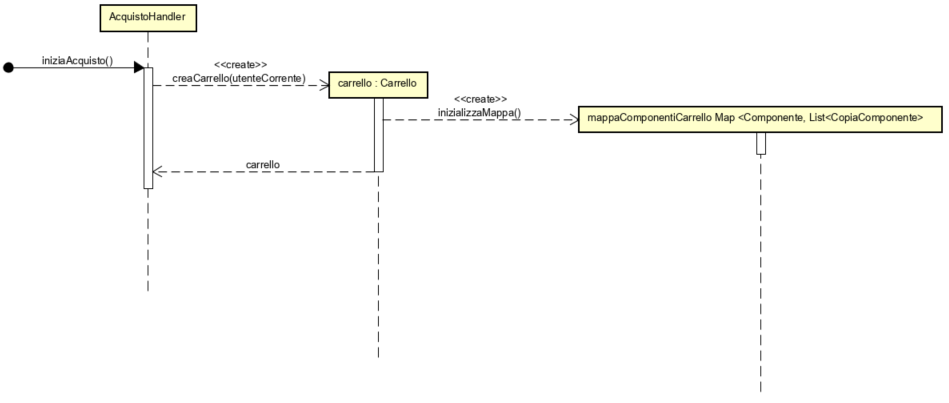
Lo scopo del suddetto caso d’uso è quello di consentire al cliente di effettuare un acquisto di una componente generica, o di una configurazione/bundle, senza dover quindi effettuare controlli sulla compatibilità delle componenti.

Inoltre tale use case prevede il controllo sulla presenza di almeno una copia dei componenti scelti prima di inserire questi ultimi nel carrello, e la rimozione di questi dalle copie componenti disponibili quando viene finalizzato l’ordine.

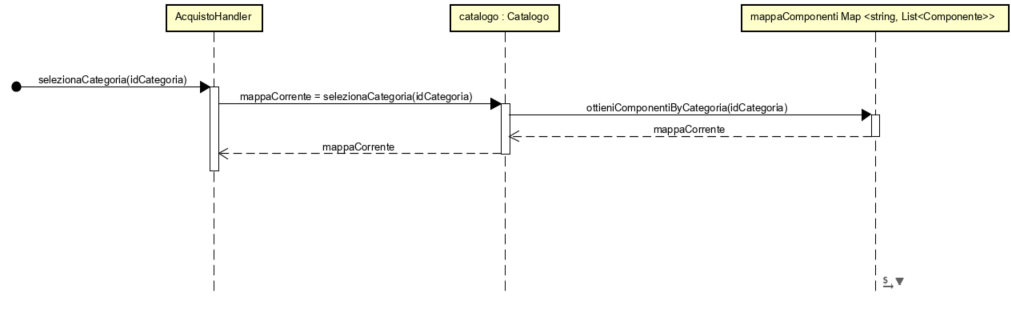
Nel caso in cui il cliente dovesse acquistare configurazioni o bundle, il controllo sulla presenza delle copie dei componenti verrà effettuato per ciascuno dei componenti che costituiscono tale configurazione/bundle: l’assenza una copia di uno qualsiasi dei componenti costituenti comporterà l’impossibilità di inserire il prodotto nel carrello.

Ecco il dettaglio dei diagrammi di sequenza per il suddetto caso d’uso:

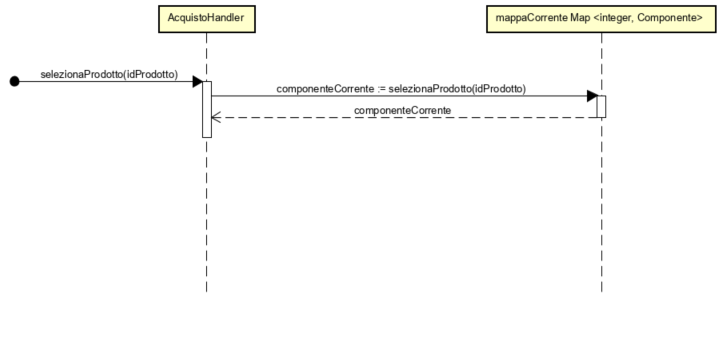
* **iniziaAcquisto**
  + Viene creata un’istanza di carrello associata al cliente corrente che sta effettuando l’acquisto



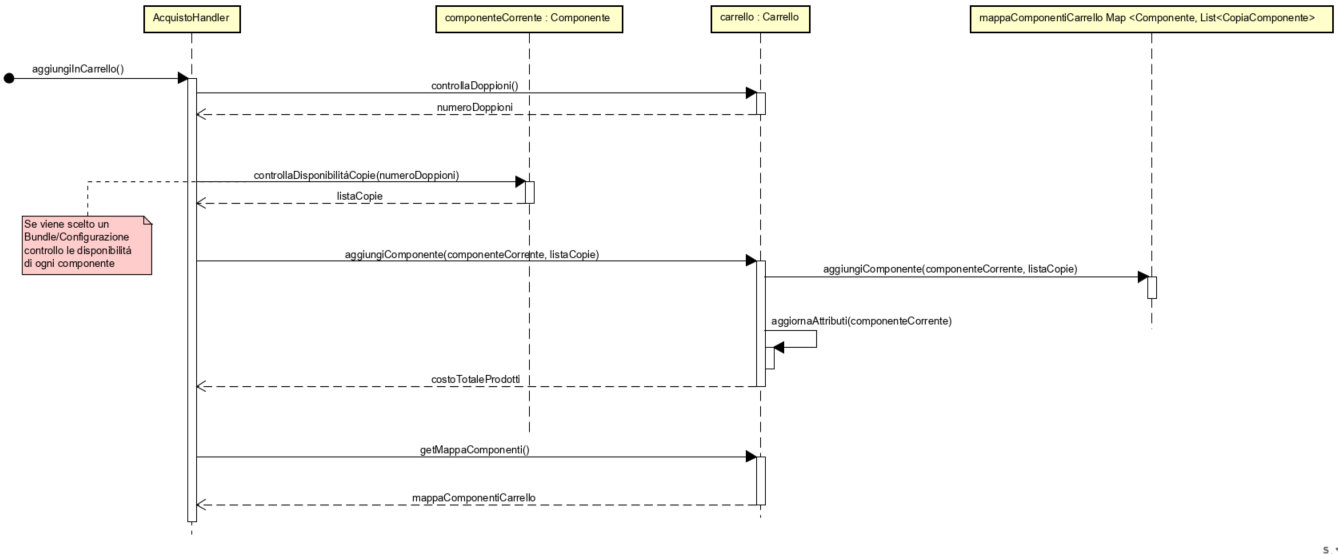
* **selezionaCategoria**
  + Viene selezionata una categoria dal cliente, e in base alla categoria scelta viene ritornata una mappa di id-componente/componente



* **selezionaProdotto**
  + Viene selezionato dalla mappa ottenuta in precedenza un prodotto, mediante l’id fornito dal cliente



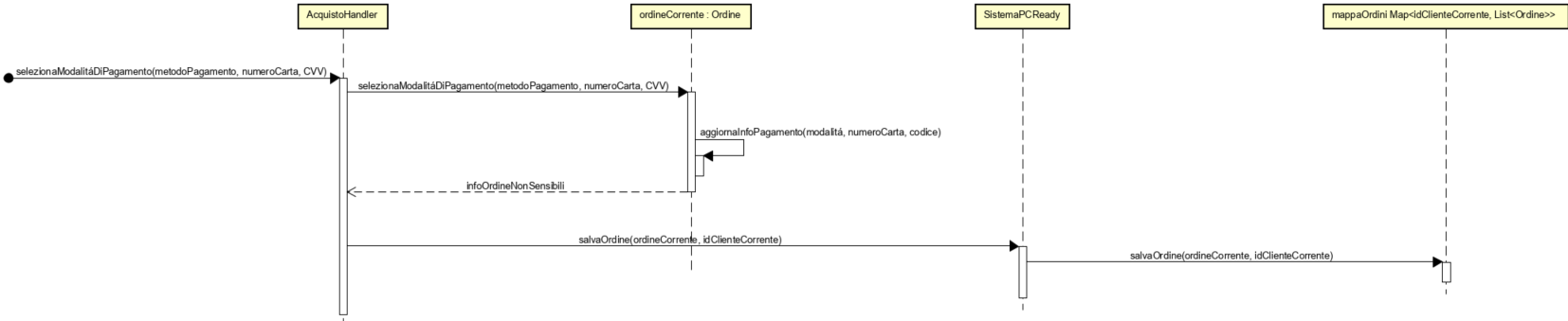
* **aggiungiInCarrello**
  + Il prodotto selezionato in precedenza dal cliente viene aggiunto al carrello, dopo un controllo sulla disponibilità delle copie del componente: tale controllo segue degli step logici ben precisi
    1. Per prima cosa viene controllata la categoria del componente corrente: se questi risulta essere una configurazione si ci deve assicurare di trovare copie componenti per ciascuno dei prodotti che la compongono
    2. Controllo doppioni: in quanto la funzione “controllaDisponibilitáCopie” ritorna la prima CopiaComponente disponibile, vi è una funzione “controllaDoppioni” che ha il compito di controllare se nel carrello è stato già selezionato lo stesso componente, ritornando un numero intero a seconda del numero delle copie già selezionate.
    3. Dunque, la funzione “controllaDisponibilitáCopie” prende come parametro un itero indicante il quantitativo di doppioni (0 equivale a nessun doppione), e recupera, se possibile, una copia corrispondente alla posizione indicata dal valore intero passato come argomento, dalla lista copie, quest’ultimo attributo di Componente.
    4. Successivamente, se viene ritornata una copia, questa viene associata al componente attuale e aggiunta al carrello mediante la funzione “aggiungiComponente”.



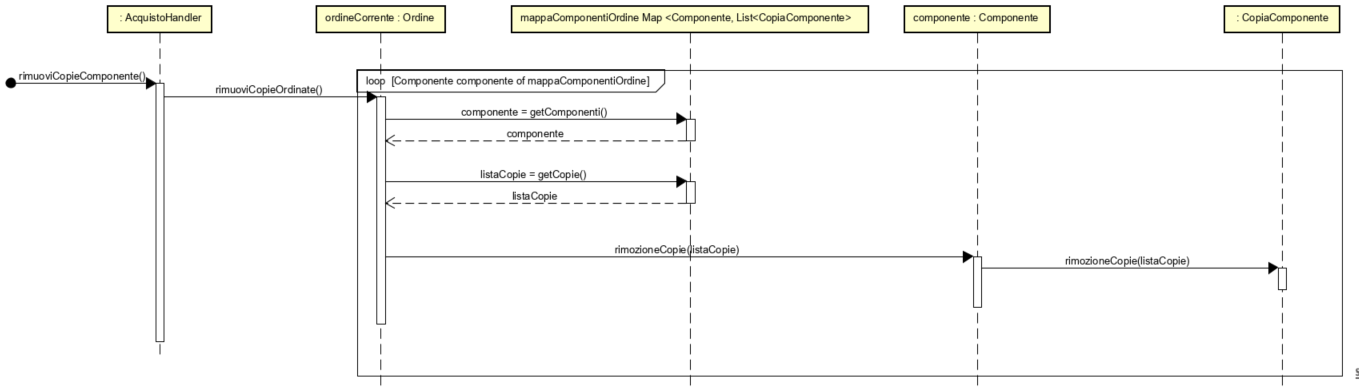
* **terminaAcquisto**
  + Tale funzione consente di inizializzare un’istanza di ordine, la quale possiede una mappa di componenti che l’utente sta acquistando, associata ad un insieme di attributi relativi al domicilio del cliente.



* **selezionaModalitáDiPagamento**
  + Il cliente mediante questa funzione è in grado di inserire la modalità di pagamento preferita, e i dati della carta che desidera utilizzare per finalizzare l’acquisto. Inoltre l’ordine verrà salvato nella memoria del sistema.



* **rimuoviCopieComponente**
  + In quanto l’acquisto è stato finalizzato, le copie che sono state acquistate dai clienti vanno rimosse in quanto non più disponibili. (L’id delle suddette copie viene comunque salvato nell’ordine effettuato, in modo tale da avere comunque traccia di ciò che è stato venduto).



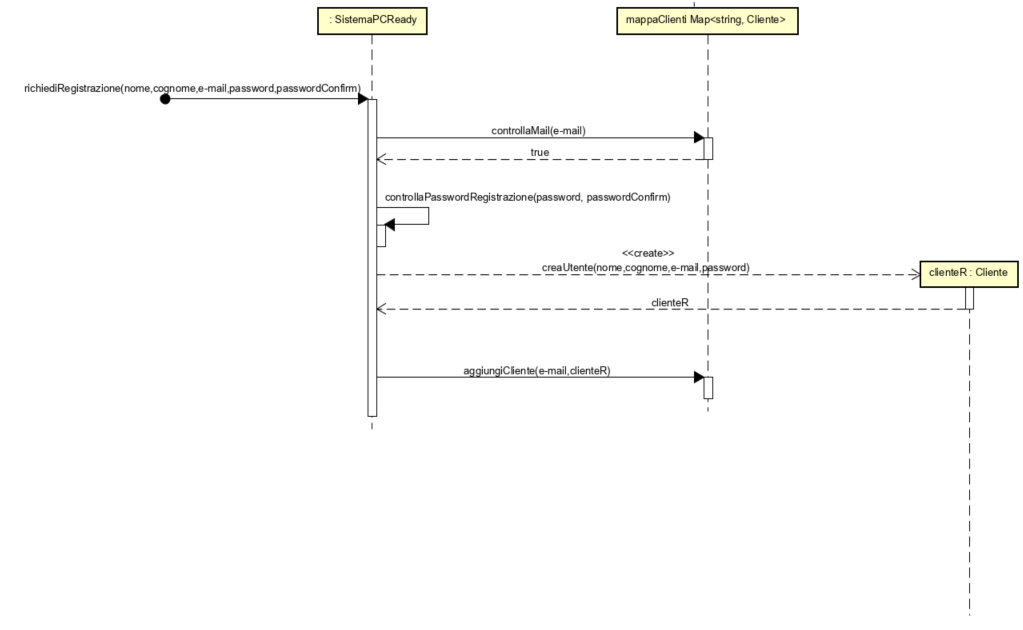
# UC6

Tale caso d’uso, nonostante esso sia un caso d’uso CRUD, risulta essere di fondamentale importanza per dar forma al progetto: difatti tale caso d’uso si occupa della registrazione del cliente, garantendo l’esistenza di utenti con una mail univoca.

Tale caso d’uso prevede un’unica funzione fondamentale detta “**richiedi registrazione**”.

Tale funzione prende in input tutti i dati del cliente, e si assicura per prima cosa che la mail fornita non sia già esistente in memoria del sistema, e successivamente si assicura che il cliente abbia inserito una password corrispondente al campo di conferma di quest’ultima.

Solo dopo aver passato con successo tali controlli verrà creata un’istanza di cliente, la quale verrà salvata nella memoria del sistema.



# UC7

Anche tale caso d’uso risulta fondamentale nonostante esso sia un caso d’uso CRUD: difatti il suddetto use case consente di autenticare un cliente o un amministratore, e di conseguenza consentire l’esecuzione (o meno) di specifiche operazioni.

Ad esempio ad un cliente sarà vietato inserire delle componenti o delle copie all’interno della memoria del sistema, in quanto tale operazione è riservata solo a quegli utenti che sono in grado di identificarsi come amministratori di sistema.

Tale caso d’uso prevede un’unica funzione fondamentale detta “**effettua login**”.

Tale funzione prende in input, oltre alla e-mail e password dell’utente, un parametro “tipologia”, in quanto si vuole comprendere se colui che sta effettuando l’accesso al sistema sia un cliente o un amministratore: sulla base di tale parametro verrà chiamata la funzione getUtente o getAmministratore, le quali effettueranno i controlli utilizzando rispettivamente una mappa di clienti e una mappa di amministratori registrati al servizio.

