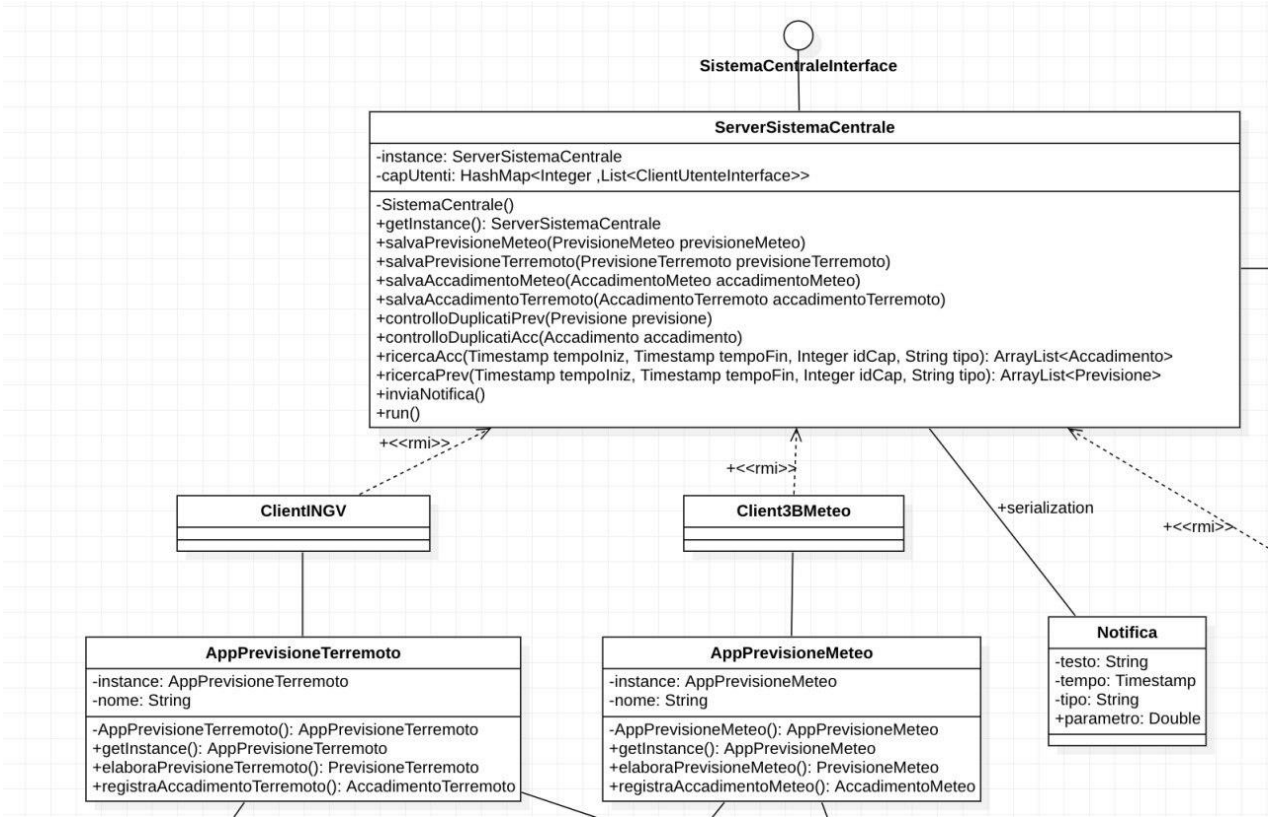
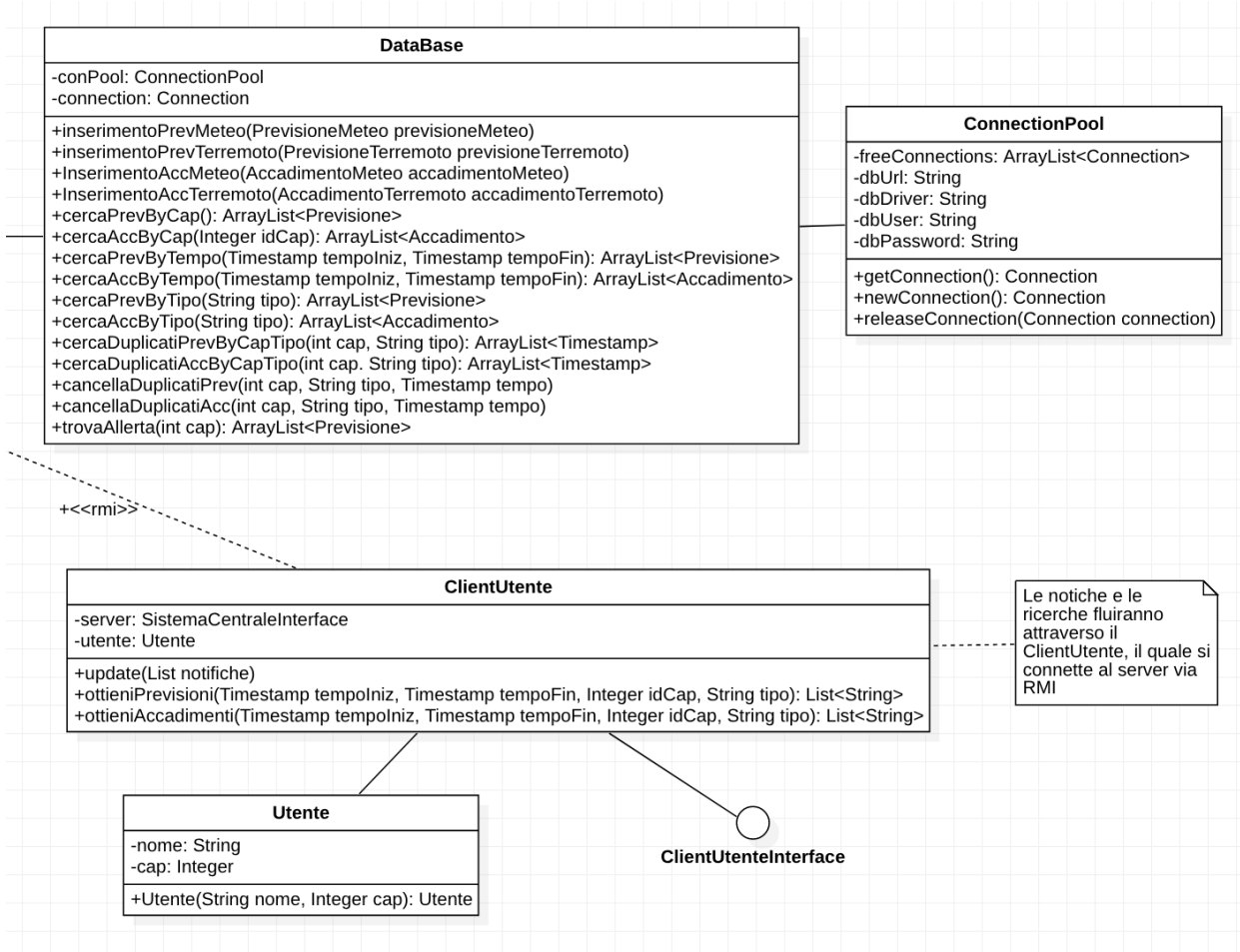
Commento UML

Class Diagram:

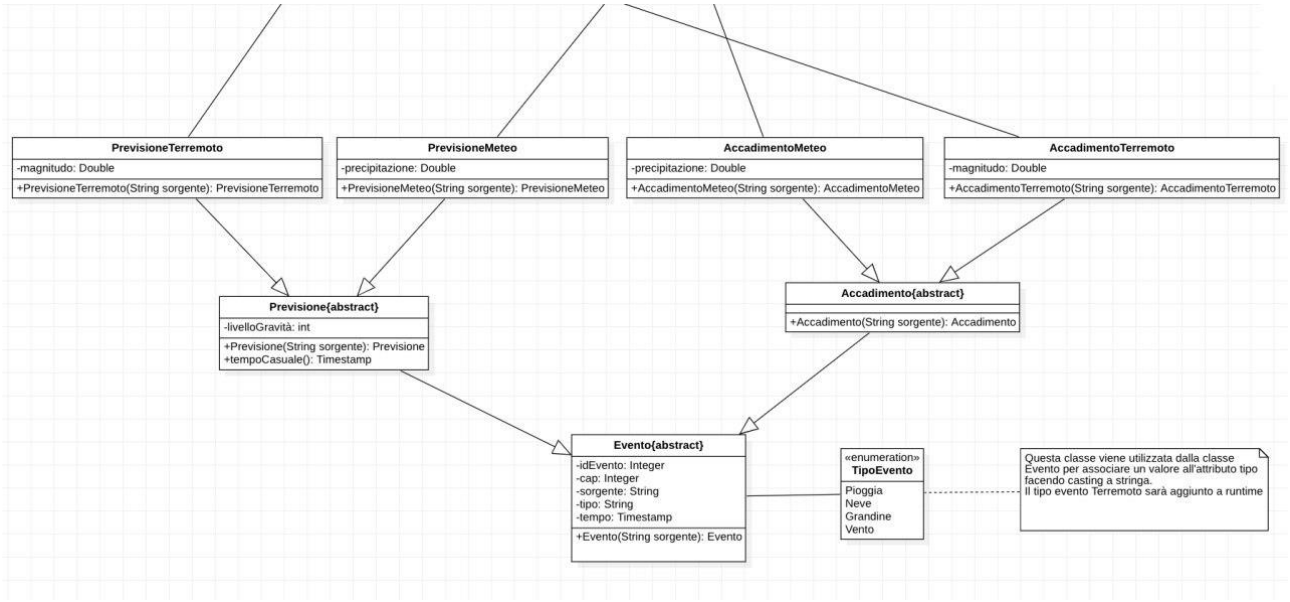


Nel class diagram possiamo vedere le classi con relativi attributi, operazioni e relazioni.
Le classi sono:
- AppPrevisioneTerremoto: è la classe che rappresenta l'applicazione che fornisce le previsioni (`elaboraPrevisioneTerremoto`) e gli accadimenti (`registraAccadimentoTerremoto`) di tipo terremoto; per farlo, chiama i costruttori delle relative classi "PrevisioneTerremoto" e "AccadimentoTerremoto"; in particolare si occupa di assegnare un livello di gravità alle previsioni in base al loro valore di magintudo.
- AppPrevisioneMeteo: è la classe che rappresenta l'applicazione che fornisce le previsioni (`elaboraPrevisioneMeteo`) e gli accadimenti (`registraAccadimentoMeteo`) di tipo meteo; per farlo, chiama i costruttori delle relative classi "PrevisioneMeteo" e "AccadimentoMeteo" assegnando poi successivamente un valore di precipitazione/velocità a seconda del tipo di evento (pioggia, neve, vento o grandine); in particolare si occupa di assegnare un livello di gravità alle previsioni in base al valore di precipitazione/velocità.
- ServerSistemaCentrale: è la classe che rappresenta il sistema che raccoglie tutte le previsioni (`salvaPrevisioneMeteo` e `salvaPrevisioneTerremoto`) e tutti gli accadimenti (`salvaAccadimentoMeteo` e `salvaAccadimentoTerremoto`) che riceve dalle applicazioni di previsione; si occupa anche di cancellare, ad ogni inserimento di una nuova previsione o di un nuovo accadimento, previsioni (`controllaDuplicatiPrev`) o accadimenti (`controllaDuplicatiAcc`) dello stesso tipo, appartenenti allo stesso cap e relativi alla stessa fascia oraria (sarebbe incoerente, ad esempio, avere due previsioni per un certo cap, di un certo tipo e relative ad una certa fascia oraria, che abbiano due livelli di gravità

completamente diversi; di conseguenza si mantiene la previsione più recente poichè considerata la più attendibile). Inoltre contiene i metodi per registrare nuovi utenti (`registraUtente`) e per elaborare per ogni cap le notifiche (`inviaNotifica`) da inviare agli utenti (salvati nella hashmap "capUtenti"). Infine contiene i metodi per la ricerca di previsioni (`ricercaPrev`) e accadimenti (`ricercaAcc`).
- Notifica: è la classe che definisce la struttura delle notifiche; ogni notifica è caratterizzata da un testo e dai parametri che descrivono l'allerta.

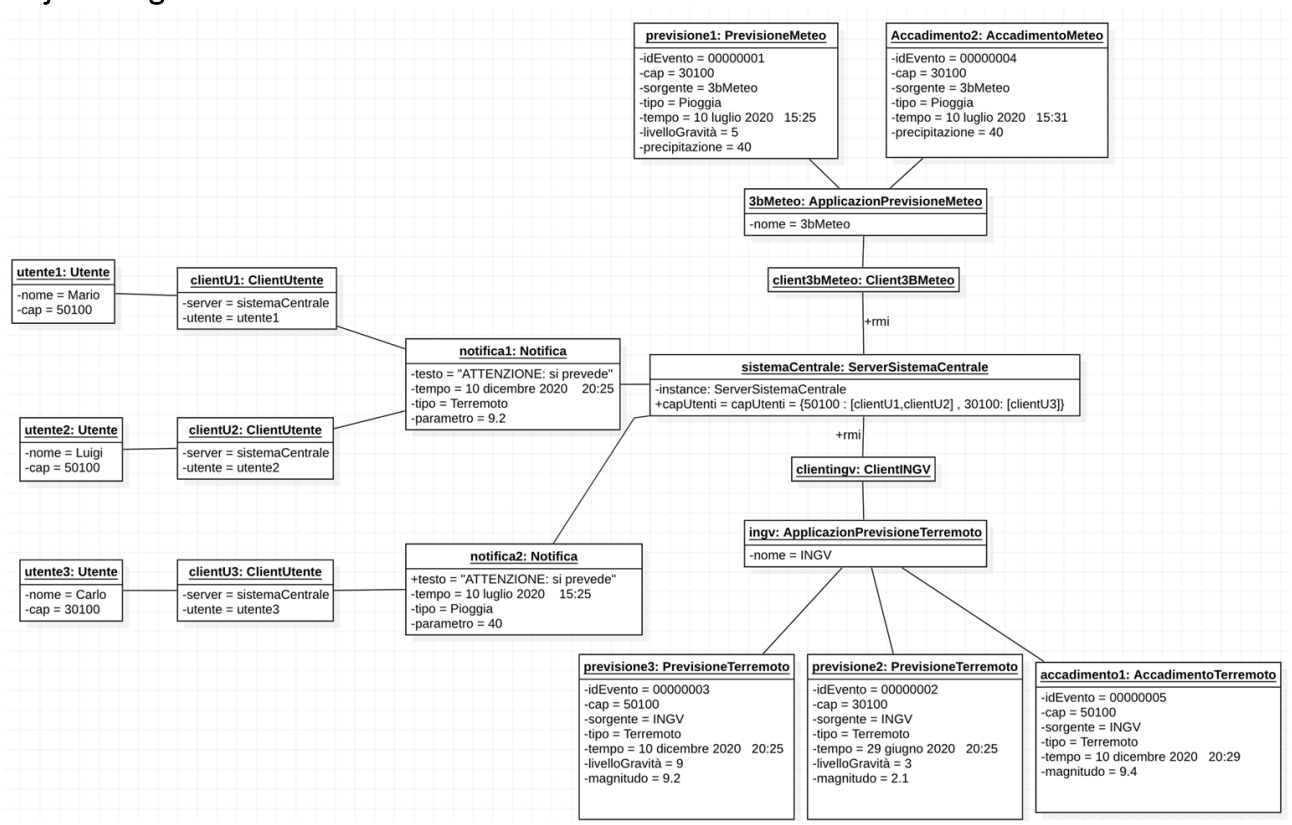


- DataBase: è la classe che contiene i metodi che si occupano di effettuare le query al database per inserire previsioni o accadimenti (`inserimentoPrevMeteo, inserimentoPrevTerremoto, inserimentoAccMeteo, inserimentoAccTerremoto`), trovare e cancellare eventuali duplicati (`cercaDuplicatiPrevByCapTipo, cercaDuplicatiAccByCapTipo, cancellaDuplicatiPrev e cancellaDuplicatiAcc`), ricercare previsioni e accadimenti secondo determinati filtri (`cercaPrevByCap, cercaAccByCap, cercaPrevByTempo, cercaAccByTempo, cercaPrevByTipo e cercaAccByTipo`) e trovare le previsioni da comunicare all'utente come allerte (`trovaAllerta`).
- ConnectionPool: è la classe che si occupa di gestire le connessioni al database (`getConnection`,`newConnection` e `releaseConnection`).
- ClientUtente: è la classe che si occupa di mostrare a video (sull'interfaccia grafica) le notifiche ricevute dal sistema centrale (`update`); si occupa inoltre di chiamare i metodi di ricerca sel sistema centrale quando viene cliccato il tasto di ricerca nell'interfaccia grafica (`ottieniPrevisioni e ottieniAccadimenti).`
- Utente: è la classe che definisce gli attributi di un utente.



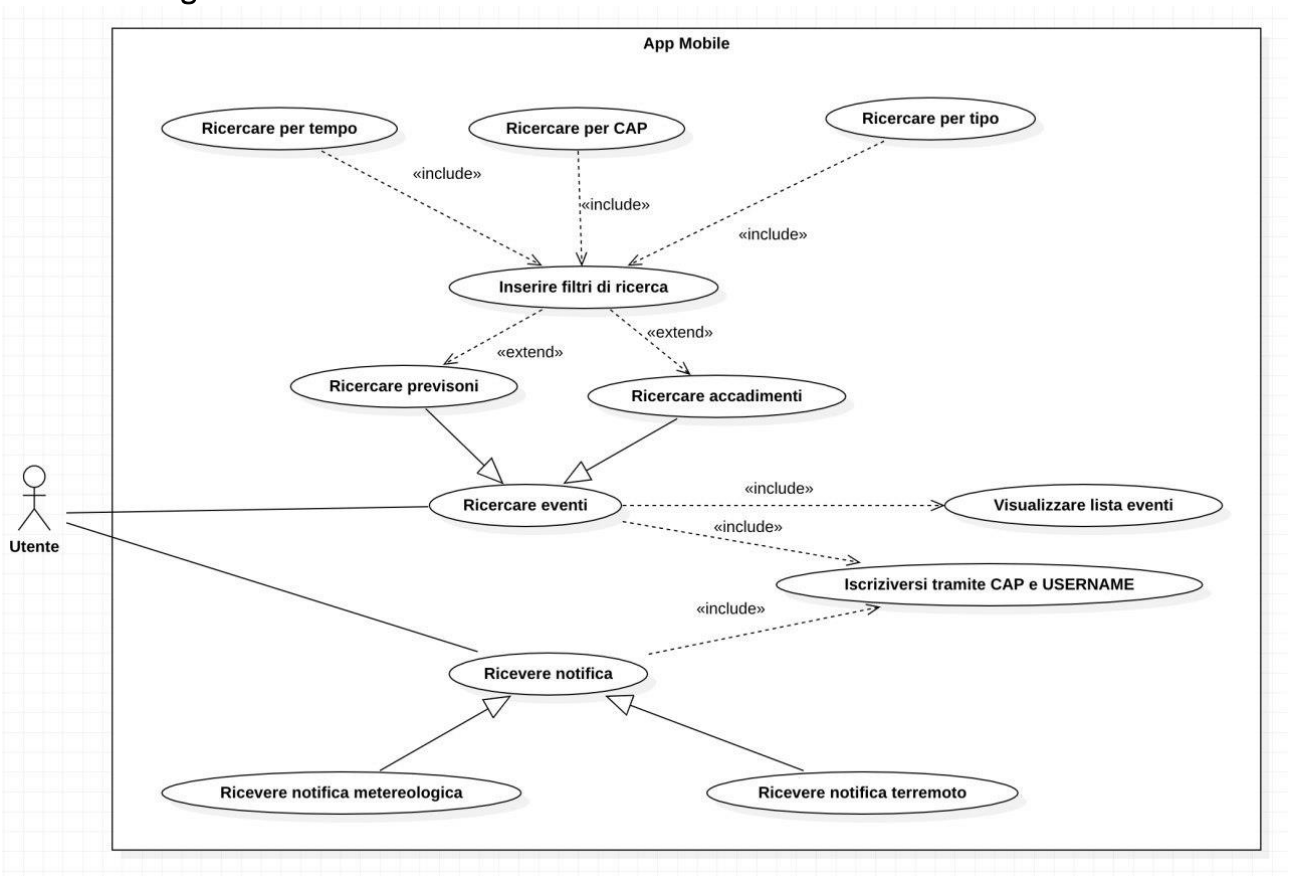
-Evento: è la classe che definisce le caratteristiche di un evento ed è quindi la superclasse di tutte le classi relative agli eventi;
-Previsione: è la classe che definisce le caratteristiche di una previsione, prelevando al contempo quelle di evento, di cui è sottoclasse.
-Accadimento: è la classe che definisce le caratteristiche di una accadimento, prelevando al contempo quelle di evento, di cui è sottoclasse.
-PrevisioneMeteo: è la classe che definisce le caratteristiche di una previsione di tipo meteo, prelevando al contempo quelle di previsione, di cui è sottoclasse.
-PrevisioneTerremoto: è la classe che definisce le caratteristiche di una previsione di tipo terremoto, prelevando al contempo quelle di previsione, di cui è sottoclasse.
-AccadimentoMeteo: è la classe che definisce le caratteristiche di un accadimento di tipo meteo, prelevando al contempo quelle di accadimento, di cui è sottoclasse.
-AccadimentoTerremoto: è la classe che definisce le caratteristiche di un accadimento di tipo terremoto, prelevando al contempo quelle di accadimento, di cui è sottoclasse.

Object Diagram:



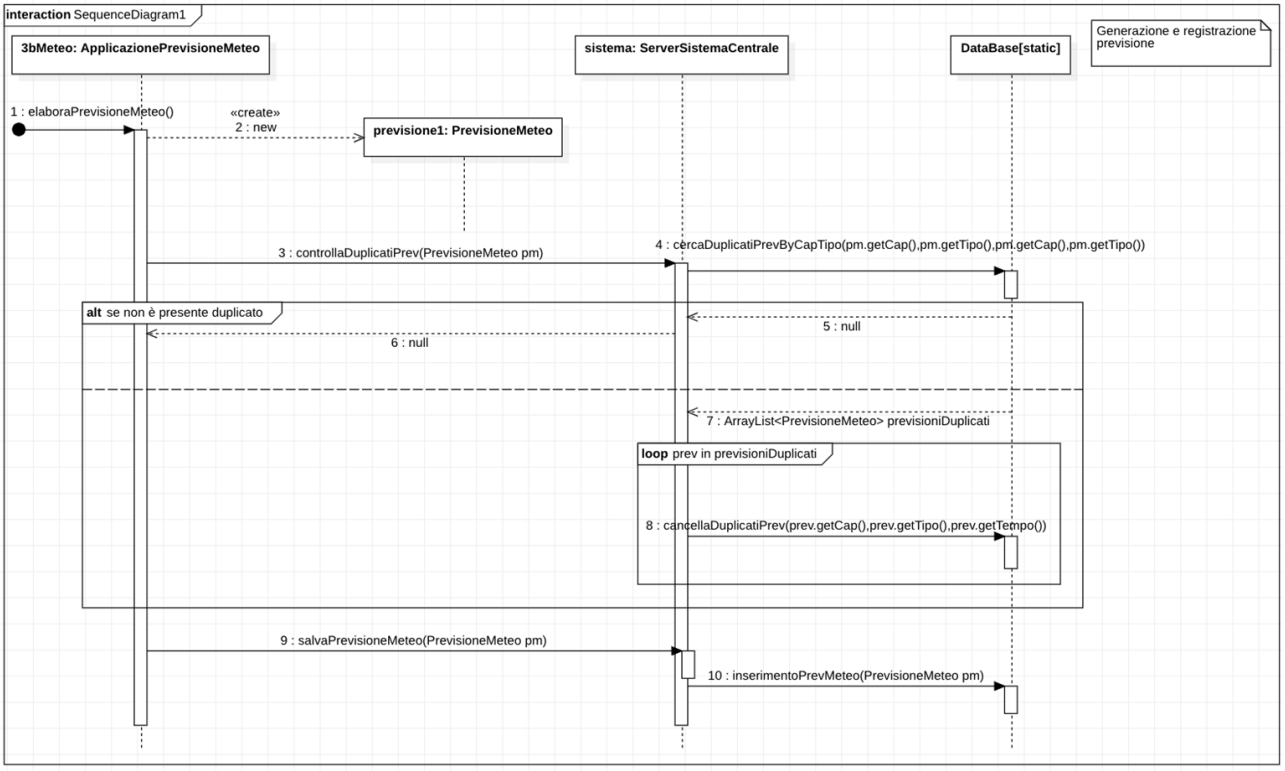
L'object diagram rappresenta un'insieme di possibili istanze delle classi.
Nell'esempio qui riportato sono presenti:
-l'istanza (unica) della classe SistemaCentrale;
-l'istanza (unica) della classe ApplicazionePrevisioneMeteo;
-l'istanza (unica) della classe ApplicazionePrevisioneTerremoto;
-le istanze (uniche) dei client relativi alle applicazioni di prevsione che comunicano col server;
-3 istanze della classe Utente, ognuna col proprio nome, di cui 2 relativi al cap 50100 e uno relativo al cap 30100;
-3 istanze della classe ClientUtente, ognuna relativa a una delle istanze della classe Utente;
-2 istanze della classe PrevisioneTerremoto, che rappresentano appunto due previsioni di tipo terremoto, di cui una relativa al cap 50100 di gravità 9 e una relativa al cap 30100 di gravità 3;
-una istanza della classe AccadimentoTerremoto, che rappresenta appunto un accadimento di tipo terremoto relativo al cap 50100;
-una istanza della classe PrevisioneMeteo che rappresenta una previsione di tipo pioggia relativa al cap 30100;
-una istanza della classe AccadimentoMeteo che rappresenta un accadimento di tipo pioggia relativo al cap 30100;
-2 istanze della classe Notifica relative alle due allerte da comunicare al cap 50100 e al cap 30100, ovvero rispettivamente la previsione di terremoto di magnitudo 9.2 e la previsione di pioggia di 40 mm/h (sono le due previsioni di massima gravità).

Use Case Diagram:



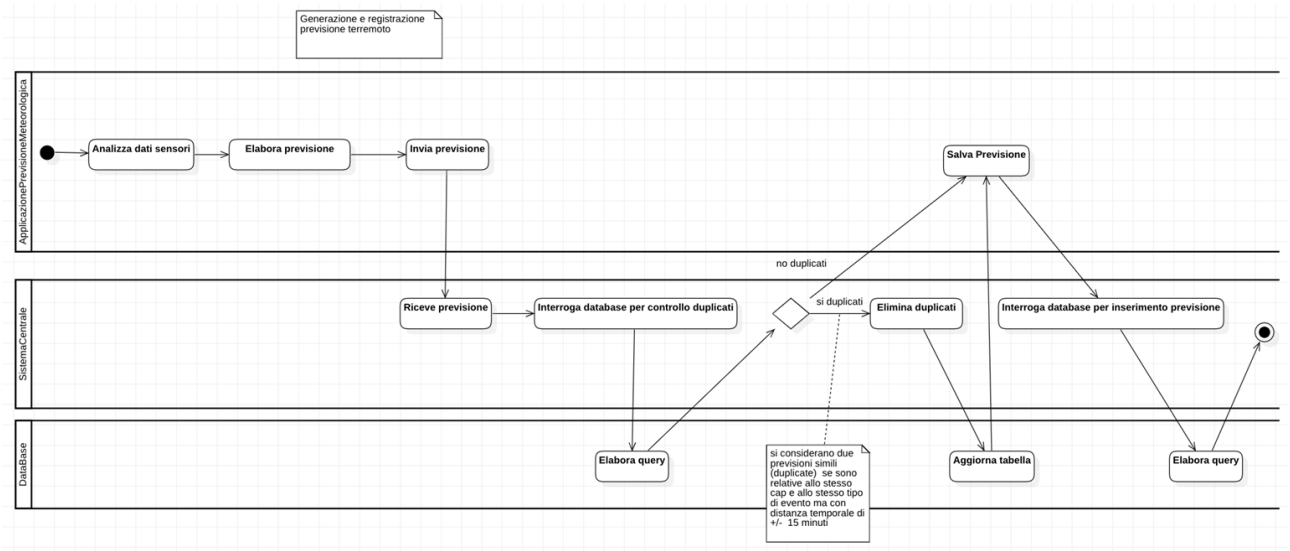
Lo use case diagram permette di definire i tipici casi d'uso del sistema, ovvero un servizio che questo offre a uno o più attori.
Nel nostro caso gli attori sono gli utenti, i quali, tramite l'applicazione, ricevono le notifiche (allerte) relative a previsioni di massima gravità e possono ricercare previsioni o accadimenti secondo determinati filtri (tempo, tipo e/o cap). Per poter compiere entrambe le operazioni, è necessario che l'utente sia iscritto all'applicazione tramite un cap e uno username

Sequence Diagram 1:



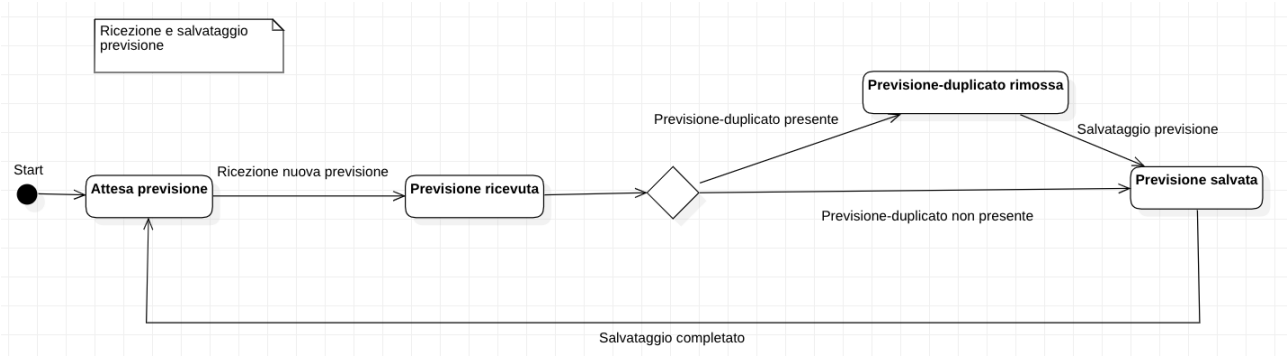
Il sequence diagram permette di mostrare i passaggi sequenziali compiuti dal programma per svolgere una determinatao procceso, mettendo in evidenza i metodi che servono e un possibile insieme di istanze delle classi a cui questi appartengono.
Nel caso in questione, il processo descritto è la generazione di una previsione di tipo meteo e il suo salvataggio nel database.
L'applicazione meteo ha il compito di generare una nuova previsione meteo chiamando il costruttore della classe omonima; poi chiama il metodo del server che controlla la presenza di eventuali duplicati della previsione appena generata, il quale a sua volta chiama un metodo della classe database che permette di prelevare dal database le previsioni che potrebbero risultare essere dei duplicati. A questo punto, se sono presenti dei duplicati, il server si occupa di eliminarli chiamando una funzione apposita dal database, altrimenti rimanda il controllo all'applicazione meteo.
Infine l'applicazione chiama la funzione del server che permette di salvare la previsione, il quale a sua volta chiama la funzione della classe database che inserisce la nuova previsione nel database.

Activity Diagram 1:



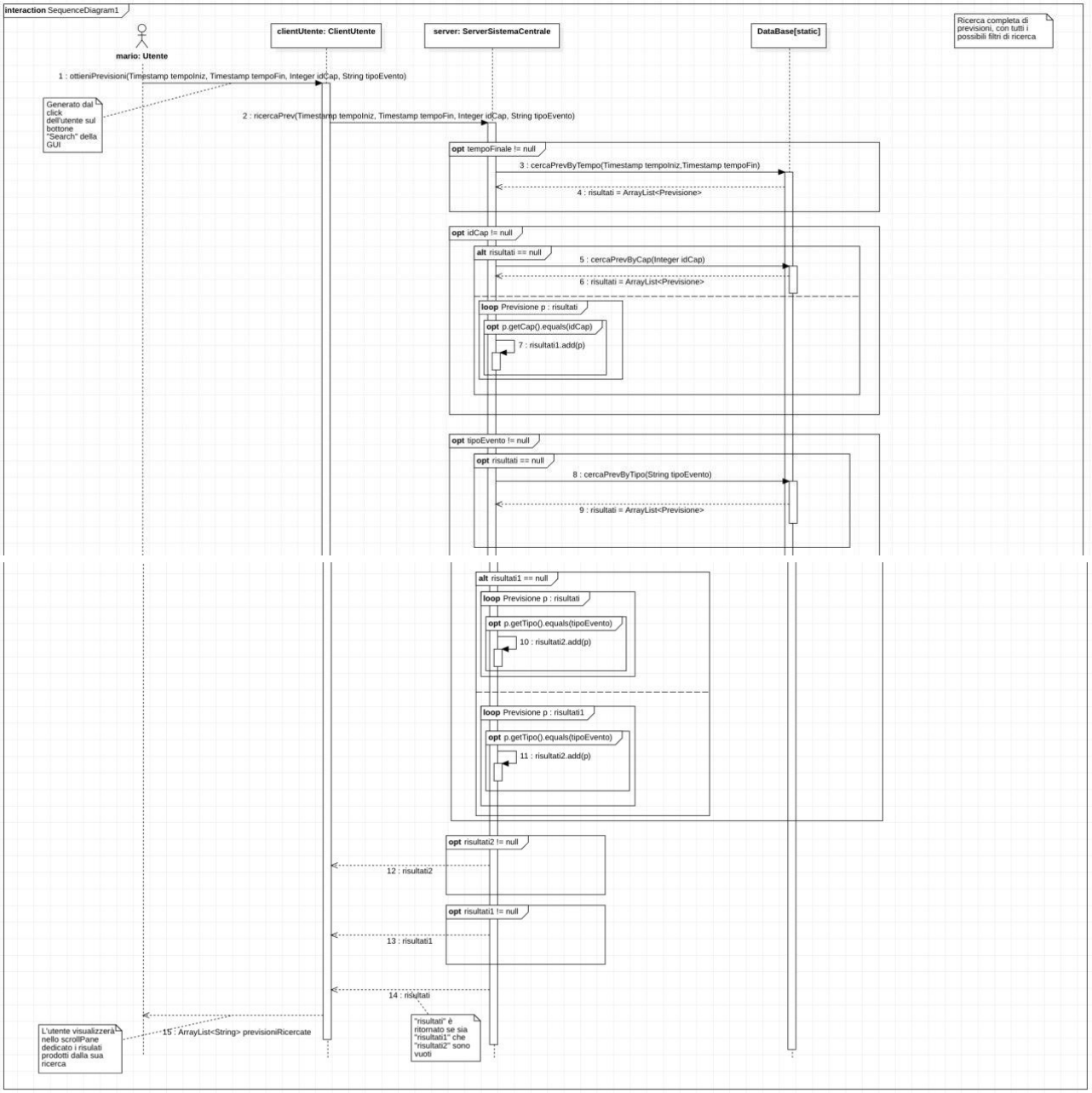
L'activity diagram permette di descrivere un processo attraverso dei grafi costituiti da nodi, che definiscono le attività, e da archi che stabiliscono l'ordine di svolgimento delle attività. Inoltre è possibile definire le classi responsabili delle varie attività tramite rettangoli, ognuno dei quali si riferisce ad una specifica classe e contiene appunto le attività ad essa delegate.
Nel caso in questione, il processo descritto riguarda la generazione di una previsione di tipo terremoto e il relativo salvataggio.
La prima attività viene svolta dall'applicazione di previsione terremoto e riguarda l'analisi dei dati che questa preleva da (ipotetici) sensori; dai dati estrapolati, l'applicazione elabora la previsione e la invia al sistema centrale. Questo riceve quindi la previsione e si rivolge alla classe database per valutare l'eventuale presenza di previsioni che risultino essere duplicati di quella ricevuta; tale classe interroga quindi il database elaborando la query necessaria e ritorna il risultato al sistema centrale. Se il risultato è vuoto, ovvero non è stato trovato nessun duplicato, allora il sistema centrale passa direttamente il controllo all'applicazione di previsione, altrimenti si occupa prima di cancellare i duplicati.
Infine l'applcazione di previsione richiede al sistema centrale il salvataggio della nuova previsione.
Quindi il processo termina aggiornando la tabella delle previsioni nel database.

State Diagram 1:



Lo state diagram descrive un processo attraverso dei grafi costituiti da nodi, che definiscono gli stati in cui si trova il sistema durante il processo, e da archi che descrivono le attività di passaggio tra uno stato e l'altro.
Nel caso in questione, il processo descritto riguarda la ricezione e il salvataggio di previsioni (meteo o terremoto).
Inizialmente il sistema è in stato di attesa di una nuova previsione; una volta ricevuta la previsione, il sistema attende l'analisi della presenza di possibili duplicati: se sono presenti duplicati, questi vengono rimossi e si attende il salvataggio della previsione, altrimenti si attende direttamente il salvataggio. Una volta salvata la previsione, il sistema torna nuovamente in stato di attesa.

Sequence Diagram 2:



In questo sequence diagram il processo descritto è la ricerca di previsioni da parte dell'utente, tenendo conto di tutte le possibili combinazioni di filtri che possono essere selezionati.

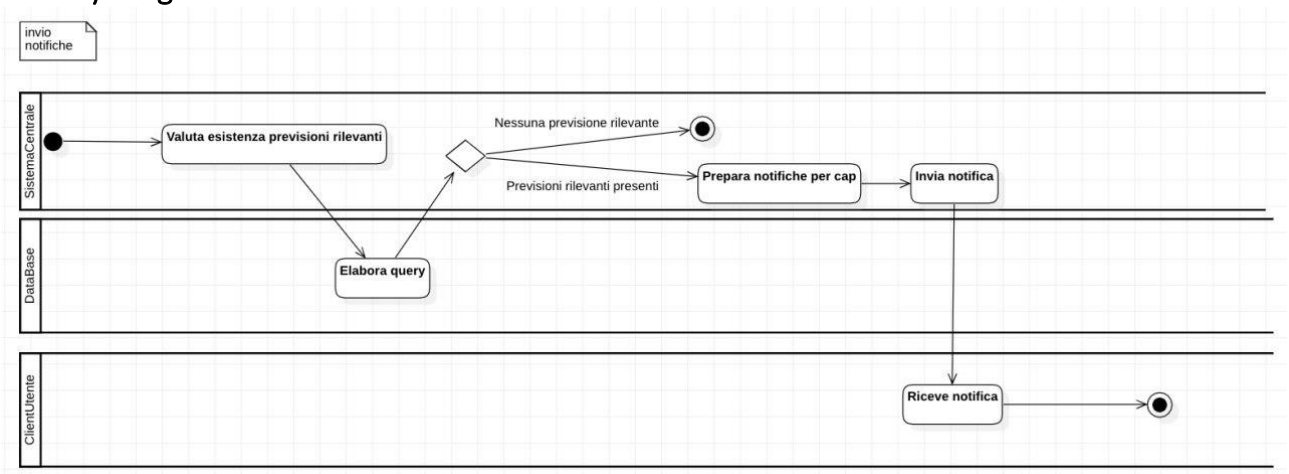
Tale processo ha inizio col click da parte dell'utente sul bottone di ricerca presente sull'interfaccia grafica; tale evento genera la chiamata di un metodo della classe client utente per ottenere le previsioni, la quale a sua volta chiama il sistema centrale comunicando i filtri inseriti dall'utente.

Il sistema centrale analizza quindi la richiesta ricevuta e, a seconda del valore dei filtri (che possono essere anche null se l'utente non ha selezionato alcun valore per un determinato filtro), decide quale precedimento seguire.

Il primo filtro che viene sempre analizzato è quello del tempo: se l'utente ha inserito una range temporale (data inizio e data fine) , il sistema centrale chiama una funzione della

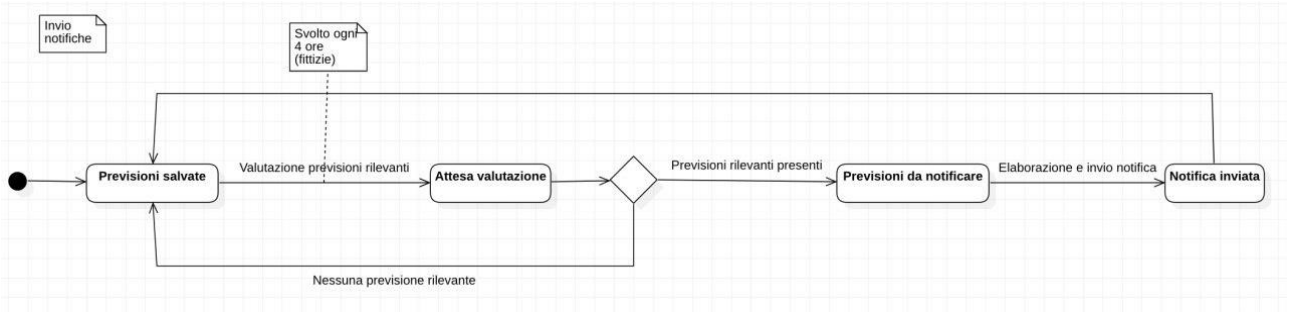
classe database che estrapola dal database le previsioni previste per quel determinato range temporale e le salva in una lista chiama "risultati", altrimenti prosegue con l'analisi dei filtri successivi.
Il secondo filtro considerato è quindi quello relativo al cap: se non è stato inserito alcun valore per tale filtro, il sistema centrale prosegue senza compiere alcuna operazione, altrimenti deve tenere in considerazioni due casi:
- se la lista "risultati" è vuota, ovvero non è stata fatta una ricerca per tempo, il sistema centrale chiama una funzione della classe database che estrapola dal database le previsioni previste per quel cap e le salva nella lista "risultati";
- se la lista "risultati" non è vuota, ovvero sono presenti nella lista delle previsioni ricavate dalla ricerca per tempo, il sistema centrale preleva da questa lista le previsioni che soddisfano anche la ricerca per cap e le salva in una nuova lista "risultati1".
A questo punto viene analizzato il terzo ed ultimo filtro, ovvero quello relativo al tipo di evento: se non è stato inserito alcun valore per tale filtro, il sistema centrale prosegue senza compiere alcuna operazione, altrimenti deve tenere in considerazione tre casi:
- se la lista "risultati" è vuota, significa che non è stata fatta una ricerca nè per tempo nè per cap e quindi il sistema centrale chiama una funzione della classe database che estrapola dal database le previsioni relative a quel tipo di evento e le salva nella lista "risultati";
- se la lista "risultati" non è vuota, ma lo è la lista "risultati1", significa che è stata già fatta una ricerca per tempo, ma non per cap; il sistema centrale quindi preleva dalla lista "risultati" solo le previsioni che soddisfano anche la ricerca per tipo e le salva nella lista "risultati1";
- se la lista "risultati" non è vuota e non lo è anche la lista "risultati1", significa che è stata fatta una ricerca sia per tempo che per cap; quindi il sistema centrale preleva dalla lista "risultati1"solo le previsioni che soddisfano anche la ricerca per tipo e le salva in una nuova lista "risultati2".
Considerati tutti i filtri, il sistema centrale analizza quindi le tre liste sopracitate partendo da "risultati2", proseguendo con "risultati1" e concludendo con "risultati": la prima lista che risulta non vuota viene ritornata alla classe client utente.
[Si noti che le liste vengono inizializzate a null]
La lista viene quindi mostrata all'utente tramite interfaccia grafica. Nel caso in cui essa sia vuota, viene mostrato un messaggio per comunicarlo.

Activity Diagram 2:



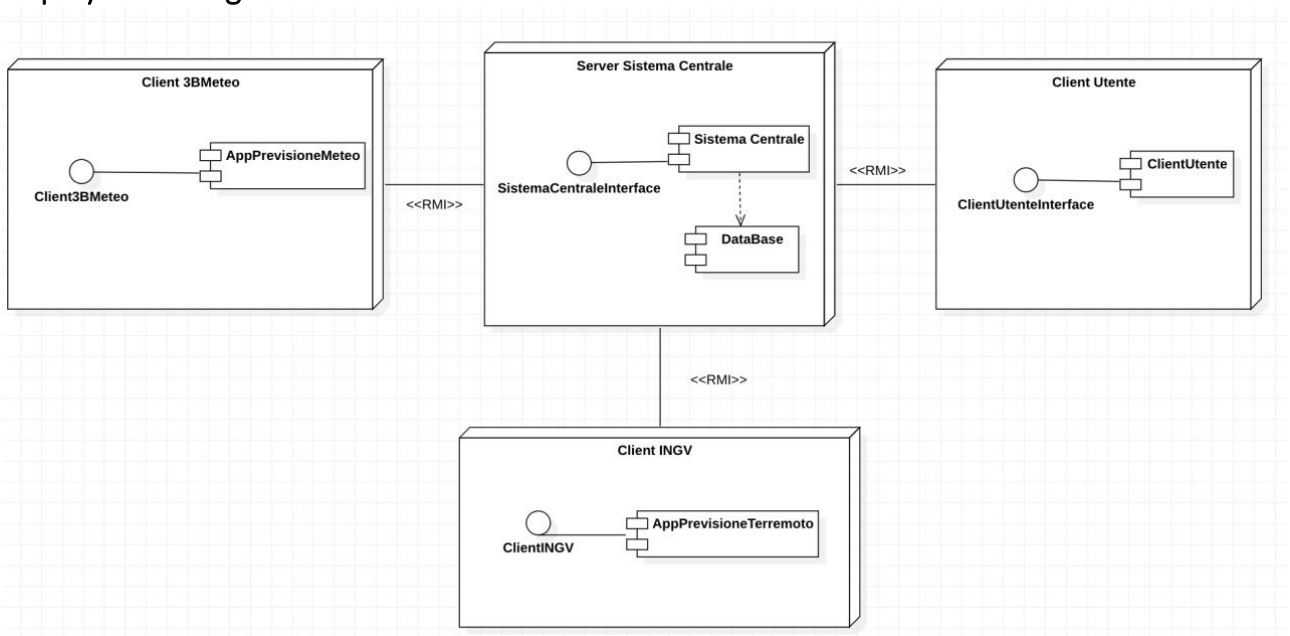
In tale activity diagram è stato descritto il processo che riguarda l'invio delle notifiche di allerta dal sistema centrale all'utente.
La prima attività viene svolta dal sistema centrale che valuta , per ogni cap, la presenza di previsioni con alto livello di gravità chiamando una funziona della classe database; tale classe interroga il database e ritorna il risultato al sistema centrale. Se il risultato non contiene nessuna previsione il processo termina, altrimenti il sistema centrale prepara le notifiche relative alle previsioni trovate e le inoltra alla classe client utente.
Quindi il processo termina.

State Diagram 2:



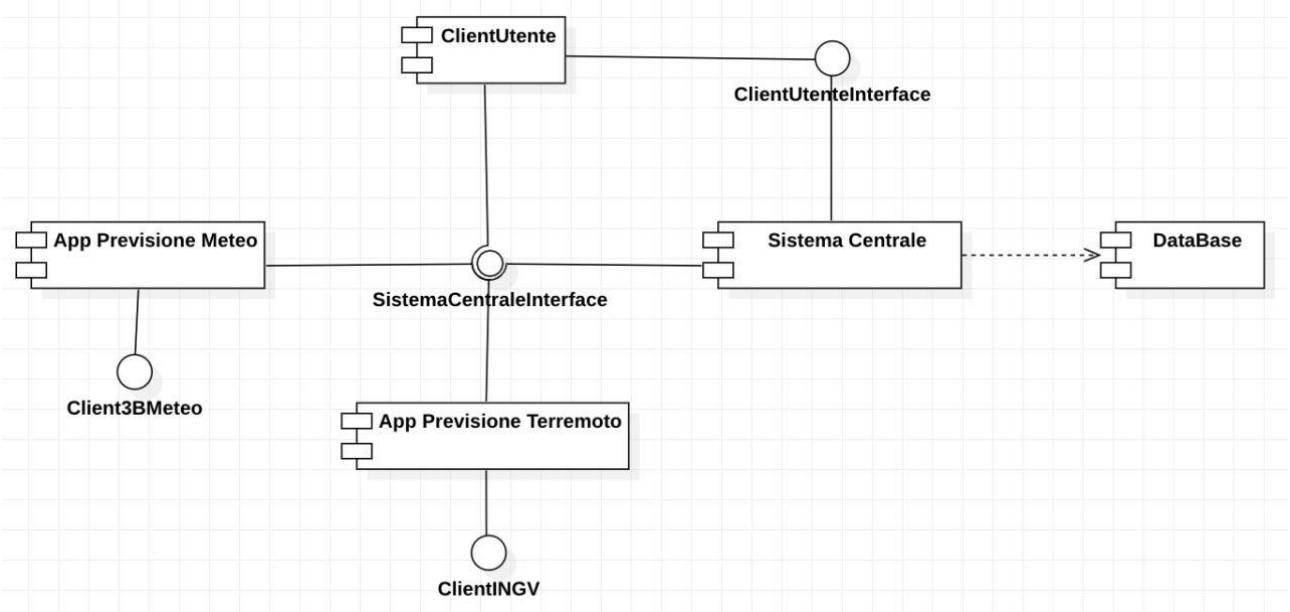
In tale state diagram il processo descritto riguarda l'invio delle notifiche.
Dopo che sono state salvate le previsioni, il sistema attende la valutazione di queste per estrapolare quelle di massima gravità (per ogni cap): se non sono presenti previsioni da allertare il sistema ritorna allo stato iniziale, altrimenti il sistema attende l'elaborazione delle notifiche e quindi il relativo invio, per poi riportarsi nello stato iniziale.

Deployment Diagram:



Il deployment diagram permette di descrivere le componenti hardware del sistema, specificando anche i loro componenti software e le loro relazioni.
Nel nostro caso le componenti hardware sono quattro:
-la componente relativa al sistema centrale che contiene il sistema centrale stesso e il database;
-le due componenti relative all'applicazione di previsione meteo e all'applicazione di previsione terremoto che contengono i rispettivi client adibiti alla comunicazione col sistema centrale (RMI);
-la componente relativa all'applicazione dell'utente che contiene il rispettivo client adibito alla comunicazione col sistema centrale (RMI).

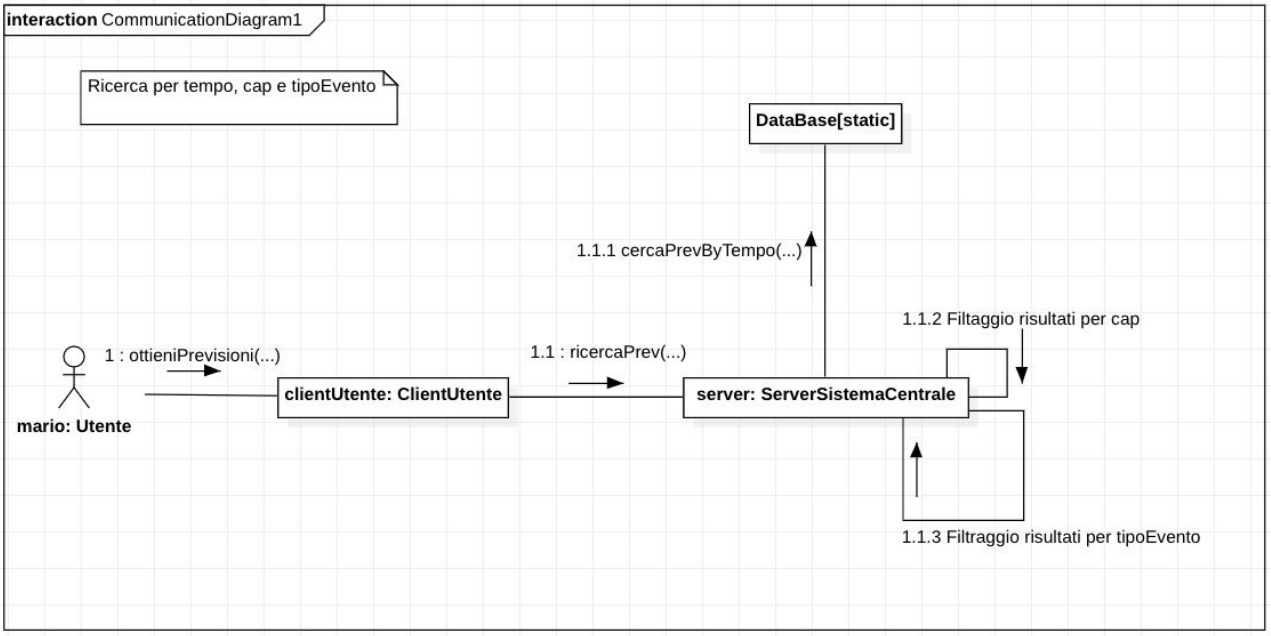
Component Diagram:



Il component diagram definisce le componenti software di un sistema con le relative interfacce, specificando le relazione che sussistono tra esse.
Nel nostro caso le componenti software sono cinque:

-la componente del sistema centrale che implementa la sua interfaccia SistemaCentraleInterface e interagisce con il componente software ClientUtente, attraverso la relativa inerfaccia, e con il database;
-la componente del database che comunica con il sistema centrale;
-le due componenti delle applicazioni di previsione (meteo e terremoto) che implementano le rispettive interfacce client e interagiscono con il sistema centrale tramite la sua interfaccia;
-la componente ClientUtente che implementa la sua interfaccia e interagisce con la componente software del sistema centrale tramite la relativa interfaccia.

Collaboration Diagram:



Il collaboration diagram mostra il flusso di interazioni e quindi le associazioni tra le istanze di diverse classi in particolari casi d'uso.
In tal caso, è stata descitta la ricerca di previsioni selezionando un valore per tutti i possibili filtri (tempo, cap e tipo).
L'interazione comincia da una istanza di Utente che richiede l'avvio della ricerca ad una istanza della classe ClientUtente attraverso l'invocazione di un metodo di questa;
ques'ultima a sua volta richiama il metodo di ricerca della (unica) istanza del sistema centrale, il quale effettua inizialmente la ricerca per tempo interrogando il database e filtra successivamente i risultati ottenuti dalla query prima per cap e poi per tipo.