Commento Java

Package

Abbiamo diviso le classi del nostro programma in 8 pacchetti distinti, a seconda della categoria di appartenenza.
Il package principale è "serverSistemaCentrale" che contiene tutte le classi relative al sistema centrale stesso e al database:
-ConnectionPool: gestisce il pool delle connessioni che vengono utilizzate dai vari processi del programma per collegarsi al database;
-DataBase: contiene tutti i metodi che servono per elaborare query al database;
-ServerSistemaCentrale: contiene le implementazioni di tutti i metodi che corrispondono alle funzionalità che deve svolgere il sistema centrale.
Vi sono poi i package relativi alle due applicazioni di previsioni meteo e terremoto.
Il primo, ovvero "trebmeteo", comprende:
-AppPrevisioneMeteo: contiene le implementazioni dei metodi che corrispondono alle funzionalità dell'omonima applicazione;
-Client3BMeteo: contiene il main che gestisce l'interazione tra l'applicazione e il sistema centrale via RMI e si occupa quindi di chiamare le funzioni necessarie (dell'applicazione o del sistema centrale).
Il secondo, ovvero "ingv", comprende analogamente:
-AppPrevisioneTerremoto: contiene le implementazioni dei metodi che corrispondono alle funzionalità dell'omonima applicazione;
-ClientINGV: contiene il main che gestisce l'interazione tra l'applicazione e il sistema centrale via RMI e si occupa quindi di chiamare le funzioni necessarie (dell'applicazione o del sistema centrale).
Un'ulteriore package è "utente" che contiene le classi:
-Utente: definisce le caratteristiche (attributi) di un utente e i relativi getter e setter;
-ClientUtente: contiene l'implementazione dei metodi che corrispondono alle funzionalità dell'utente.
È presente poi il package "eventi" che comprende tutte le classi relative ai possibili eventi, ovvero:
-Evento: definisce gli attributi di un evento e contiene i relativi getters e setters e il proprio costruttore; è la superclasse di tutte le classi appartenenti a questo package;
-Accadimento: contiene il proprio costruttore ed è superclasse delle classi AccadimentoMeteo e AccadimentoTerremoto ;
-AccadimentoMeteo: definisce l'attributo aggiuntivo "precipitazione", i relativi getter e setter e il proprio costruttore;
-AccadimentoTerremoto: definisce l'attributo aggiuntivo "magnitudo", i relativi getter e setter e il proprio costruttore;
-Previsione: definisce l'attributo aggiuntivo "livelloGravità", i relativi getter e setter e il proprio costruttore e contiene un metodo, richiamato dal costruttore, per la generazione casuale del tempo in cui si prevede la previsione; è superclasse delle classi PrevisioneMeteo e PrevisioneTerremoto;
-PrevisioneMeteo: definisce l'attributo aggiuntivo "precipitazione", i relativi getter e setter e il proprio costruttore;
-PrevisioneTerremoto: definisce l'attributo aggiuntivo "magnitudo", i relativi getter e setter e

il proprio costruttore.
Abbiamo definito anche un package "condivise" che contiene le classi che possono essere utilizzate da altre classi appartenenti a più package diversi; tali classi sono:
-SistemaCentraleInterface: è l'interfaccia che contiene la dichiarazione dei metodi che vengono implementati dalla classe ServerSistemaCentrale;
-ClientUtenteInterface: è l'interfaccia che contiene la dichiarazione dei metodi che vengono implementati dalla classe ClientUtente;
-ListaCap: è una classe enumeration che contiene la lista dei cap a cui il sistema manda le notifiche di allerta, un metodo per prelevare randomicamente uno dei cap e un metodo per prelevare l'intera lista;
-TipoEvento: è una classe enumeration che contiene i possibili tipi di evento meteorologico che sono gestiti dal sistema, un metodo per prelevare un tipo di evento randomicamente e un metodo per prelevare l'intera lista;
-Notifica: definisce gli attributi di una notifica e contiene i relativi getters e setters e il proprio costruttore;
-RandomGaussiano: è la classe che contiene i metodi per la generazione di valori di precipitazione (per gli eventi di tipo meteo) o di magnitudo (per gli eventi di tipo terremoto) secondo una distribuzione gaussiana.
È presente inoltre il package "clientGUI" che contiene l'omonima classe che costituisce l'interfaccia grafica dell'applicazione dell'utente.
Infine abbiamo creato un package "test" che contiene due classi di test JUnit [vedi paragrafo successivo].

Test JUnit

Per testare il nostro programma, abbiamo creato due classi che verificano il corretto funzionamento dei metodi della classe database che gestiscono gli eventi.
Il primo test riguarda la gestione di una previsione meteo.
Viene quindi inizialmente generata una istanza della classe "PrevisioneMeteo", assegnandole determinati valori di test per ogni attributo, e si verifica che l'istanza creata contenga effettivamente tali valori scelti. L'istanza viene quindi inserita nel database.
Si verifica poi il corretto funzionamento di tutti i metodi possibili che riguardano la ricerca di previsioni; per farlo viene chiamato ogni metodo, passando i parametri necessari relativi alla nostra previsione di test, e si controlla per ciascuno che la previsione di test sia presente tra quelle trovate.
Lo stesso controllo viene effettuato sul metodo che ricerca le previsioni di massima gravità da notificare; abbiamo infatti assegnato un valore alto al parametro di test che riguarda il livello di gravità della previsione, in modo che questa risulti nella lista delle previsioni da allertare.
Infine si cancella la previsione di test dal database utilizzando la funzione di cancellazione dei duplicati (per le previsioni) e si controlla che sia effettivamente stata eliminata utilizzando (arbitrariamente) la funzione che ricerca le previsioni tramite cap e verificando che essa non sia presente tra i risultati della ricerca.
Il secondo test è analogo, ma riguarda la gestione di un accadimento di tipo terremoto.
Come nel primo caso, viene generata una istanza della classe "AccadimentoTerremoto", le vengono assegnati dei valori di test e si verifica che tali valori siano effettivamente associati

all'istanza appena creata.
Si inserisce quindi l'accadimento di test nel database per poter verificare il corretto funzionamento dei metodi della classe database per la ricerca di accadimenti, così come fatto nel primo test.
Infine si cancella l'accadimento dal database tramite la funzione di cancellazione dei duplicati (per gli accadimenti) e si controlla che sia stato effettivamente eliminato utilizzando (arbitrariamente) la funzione che ricerca gli accadimenti tramite cap e controllando che esso non sia presente tra i risultati della ricerca.