La prima versione del class diagram(del D2) è stato ricavato mediante un processo di forward engineering sullo use case diagram(per quanto riguarda le classi), i metodi sono stati ricavati maggiormente dal component diagram. La seconda versione è stata ricavata dalle modifiche apportate durante la scrittura del codice.

Questo modello si divide in quattro package:

* **Users**: contiene la classe astratta Utente che generalizza Admin e Gestore. Possiedono gli attributi necessari per l’accesso nel sistema quali username e password, uno che indica il tipo di utente(livelloDettaglio), e uno esclusivo della classe Gestore(IDZona) che indica la zona di competenza(edificio, distretto o città). I metodi forniti sono quelli basilari per restituire e modificare tali attributi, e i costruttori;
* **GUI**: le classi qui presenti non hanno attributi, ma solo metodi statici messi a disposizione degli utenti. PaginaIniziale, usata da tutti i tipi di utenti, permette l’accesso e l’uscita dal sistema e la modifica delle proprie credenziali. DasboardAdmin, utilizzabile esclusivamente dagli utenti admin, mette a disposizione tutto il necessario per la gestione backend (inserimento, cancellazione e modifica di dettagli degli utenti e degli elementi mappati contenuti nel database). DasboardGestore si occupa della visualizzazione dei dati monitorati, la notifica di pericoli e la modifica del livello di dettaglio, ed è utilizzabile solo dagli utenti gestori;
* **Mappa**: contiene la classe astratta ElementoMappa che generalizza Sensore e Città, che a sua volta generalizza Distretto, Edificio e Area. Gli attributi comuni a tutti sono ID per l’identificazione e alert per rappresentare lo stato di allerta (stato di funzionamento nei sensori). Nella classe Sensore vengono aggiunti: tipo (indica l’unità di misura); variabileAmbientale (ultimo valore registrato); massimale (indica il valore soglia); frequenzaInvio (per regolare la frequenza di invio di segnali); ultimoInvio (indica l’ora in cui è stato ricevuto l’ultimo segnale). Nella classe Città si aggiunge il solo numElementi che rappresenta il numero di elementi contenuti. Anche qui, tutti i metodi sono i costruttori e quelli basilari per restituire e modificare tali attributi, e in più ci sono aggiungiElementi e rimuoviElemento della classe Città, che servono a modificare la lista degli elementi contenuti. La classe più grande contiene elementi della classe più piccola, e in questo caso l’ordine di grandezza (crescente) è: Sensore, Area, Edificio, Distretto, Città;
* **Server**: contiene le classi Segnale e Database. Il primo viene generato nel momento in cui il server riceve il segnale inviato da un sensore e inizia il suo processamento. Possiede tre attributi: IDSensore (ID del sensore che ha inviato il segnale), variabileAmbientale (valore misurato), alert (che indica lo stato di funzionamento del sensore); e i soli metodi per restituire i suoi attributi, infatti, i dati del segnale servono solo per aggiornare lo stato del sensore associato, e una volta fatto ciò, vengono eliminati. Database, invece, non ha attributi e definisce i metodi statici per (elenco in ordine cronologico di esecuzione): leggere i dati di un segnale e utilizzarli per aggiornare quelli del sensore inviante (aggiornaDatiSensore); si controlla se i nuovi dati producono anomalie o errori a livello di sensore (RilevaAnomalieOErrore); in funzione del risultato del controllo precedente la frequenza di invio viene modificata (modificaFrequenza); se si trovano anomalie o errori si calcola la presenza di pericoli ai livelli di area, edificio e distretto che contengono il sensore (calcoloPericolo). Una volta compiuta questa serie di azioni, il segnale processato viene eliminato e si passa al successivo.