Analisi e Progettazione del Software

Prova scritta del 21 giugno 2019

Si vuole progettare una classe C++ per un sistema di gestione di droni per il controllo di sensori a terra. Nel sistema sono presenti un insieme di sensori dislocati nel territorio, ognuno con le sue coordinate (2 valori reali, in metri, rispetto ad un'origine convenzionale). Inoltre, è presente un insieme di droni che si muovono nel territorio, per i quali si assume, per semplicità, che l'altezza sia trascurabile, quindi anche la loro posizione è espressa tramite due coordinate.

Un sensore può essere "agganciato" ad un drone se questo è ad una distanza (euclidea, sul piano) minore o uguale dello specifico raggio di azione del sensore stesso. Ciascun drone ha un numero massimo di sensori che possono essere agganciati ad esso. Se un sensore è agganciato ad un drone e quest'ultimo si sposta a distanza maggiore del raggio di azione del sensore, allora il sensore si sgancia e si riaggancia al drone ad esso più vicino che lo abbia nel suo raggio d'azione e che abbia disponibilità libera (altrimenti il sensore rimane scollegato). Inoltre, quando un drone si sposta tutti i sensori del sistema scollegati, che lo abbiano nel raggio di azione nella nuova posizione si collegano ad esso (fino alla sua disponibilità).

Le operazioni principali sono le seguenti:

$InserisciSensore(s : Sensore, x : real, y : real)$
Il sensore s viene registrato tra i sensori del sistema in posizione (x, y). Il sensore si aggancia al drone più vicino, se possibile.
Precondizioni: Il sensore non è già registrato nel sistema.

$EliminaSensore(s : Sensore)$
Il sensore s viene eliminato dai sensori del sistema. Se il sensore era agganciato ad un drone, questo viene scollegato. *Precondizioni:* Il sensore è già registrato.

$InserisciDrone(d : Drone, x : real, y : real)$
Il drone d viene registrato nel sistema in posizione (x, y). Tutti i sensori presenti nel sistema scollegati che siano nel suo raggio di azione si collegano al drone (fino a raggiungere la sua disponibilità).
Precondizioni: Il drone d non è già registrato.

$SpostaDrone(d : Drone, x : real, y : real)$
Il drone d si sposta di (x, y) metri dalla sua posizione precedente. Tutti i sensori ad esso collegati che non lo hanno più nel raggio di azione, si collegano ad un altro drone (se possibile). Tutti i sensori scollegati si collegano a d, se possibile.
Precondizioni: Il drone d è già registrato.

Esercizio 1 (punti 5) Si disegni il diagramma UML delle classi per l'applicazione.

Esercizio 2 (punti 5) Si scriva la definizione della classe `Gestore` e delle altre classi che compongono il diagramma UML.

Esercizio 3 (punti 10) Si scrivano le definizioni dei metodi della classe `Gestore` che corrispondono alle operazioni sopra elencate e i selettori che si ritengono opportuni. Si gestiscano le precondizioni tramite il lancio dell'eccezione `invalid_argument`. Si definiscano i metodi (modificatori e selettori) delle altre classi del diagramma UML.

Esercizio 4 (punti 4) Si scriva l'operatore di output della classe `Gestore`, in modo che stampi anche tutti i dati ad essa collegati.

Esercizio 5 (punti 6) Si scriva una funzione esterna (non *friend*) che riceva come parametro un oggetto della classe `Gestore`, un drone (supposto già registrato) ed un vettore di coppie di valori reali, che rappresentano una sequenze di spostamenti per il drone stesso. La funzione deve eseguire gli spostamenti del vettore e deve restituire il numero totale di sensori (inclusi i duplicati) che il drone ha servito in totale nelle posizioni visitate (esclusa quella di partenza, inclusa quella di arrivo). A questo scopo, si definiscano degli opportuni selettore nella classe `Gestore`.