Autorità Garante
della Concorrenza e del Mercato
Autorità Nazionale Anticorruzione

Concorso pubblico, per titoli ed esami, a 2 posti nella qualifica di funzionario in prova, uno nel ruolo della carriera direttiva dell'Autorità Garante della Concorrenza e del Mercato (VI° livello della scala stipendiale) e uno nel ruolo dell'Autorità Nazionale Anticorruzione (Categoria A, parametro retributivo F1), per lo svolgimento di attività di indagine, progettazione, sviluppo e di *reverse engineering* di software, algoritmi e data base. (G.U. - IV^ Serie speciale - Concorsi ed esami, n. 56 del 25/7/2017).

Prova tecnico-pratica scritta del 20 marzo 2018

TRACCIA n° 1*(Traccia estratta)

Requisiti. L'applicazione da progettare riguarda la parte modellistica di un videogioco (non l'interfaccia). Ad ogni livello del gioco è associato un valore di energia di accesso, un valore di energia di uscita, una mappa, un insieme di personaggi. La mappa ha un nome, un insieme di punti di interesse e un riferimento ad un file che ne descrive le caratteristiche grafiche (i cui dettagli non interessano). I punti di interesse hanno un identificatore e sono legati tra loro da una relazione di adiacenza. I personaggi hanno un nome che li identifica, un riferimento ad un file che ne descrive le caratteristiche grafiche (i cui dettagli non interessano), un livello di energia, e sono posizionati in uno dei punti di interesse della mappa. I personaggi sono divisi in 3 categorie: i vampiri, i licantropi e i mutaforma. I licantropi sono organizzati in tribù e ognuno di loro appartiene esattamente ad una tribù. Ogni tribù ha un nome che la identifica e contiene almeno 10 licantropi. I mutaforma possono assumere la forma di un altro personaggio, purché esso sia un vampiro o un licantropo, prendendone tutti i poteri e, durante il gioco, possono cambiare forma secondo le proprie necessità.

Il comportamento dei personaggi dipende dalla loro categoria.

- Ogni *vampiro* può fare le seguenti 2 azioni. *Vaga*, che lo sposta da un punto di interesse ad un altro (in modo randomico). *Attacca* che può essere eseguita solo quando il vampiro si trova nello stesso punto di interesse di un licantopo o di un mutaforma, che a sua volta non ha assunto le sembianze di un vampiro e che lo porta allo *scontro*. Non tutte le sequenze di azioni sono disponibili. In particolare, dopo *Attacca* l'unica azione disponibile è *Vaga*.

- Ogni *licantropo* può fare le seguenti 3 azioni. *Vaga*, che lo sposta da un punto di interesse ad un altro (in modo randomico). *Avvicinati* alla tribù, che lo sposta dal suo corrente punto di interesse ad uno dei punti di interesse adiacenti dove sono più presenti altri membri della sua tribù (si noti che, se non ci sono membri della sua tribù nei punti di interesse adiacenti, questa azione si comporta come *Vaga*). *Difenditi*, che lo porta allo *scontro* con un eventuale vampiro che lo ha appena attaccato: questa azione può essere adottata solo in caso di attacco da parte di un vampiro e in tal caso è l'unica azione disponibile ed è obbligata.

- Ogni *mutaforma* può fare le seguenti 3 azioni. *Vaga*, che lo sposta da un punto di interesse ad un altro (in modo randomico). *PrendiForma*, che gli fa prendere la forma di un vampiro o un licantropo che si trova nello stesso punto di interesse e può essere adottata solo se un vampiro o un licantropo sono presenti nello stesso punto di interesse. *Difenditi*, che lo porta allo *scontro* con un eventuale vampiro che lo ha appena attaccato: questa azione può essere adottata solo in caso di attacco da parte di un vampiro e in tal caso è l'unica azione disponibile ed è obbligata.

Gli scontri sono decisi in questo modo. Se un vampiro attacca un licantropo isolato (che non ha altri membri della sua tribù nel suo punto di interesse), oppure un mutaforma senza alcuna forma o con la forma di un licantropo isolato, allora vince. Se un vampiro attacca un licantropo non isolato (che ha altri membri della sua tribù nel suo punto di interesse) o un mutaforma con la forma di un licantropo non isolato, allora l'avversario vince. Quando un personaggio vince incrementa la sua energia di una unità mentre quando perde, diminuisce la sua energia di una unità.

Siamo interessati alla seguente attività principale. L'attività prende in input un livello del gioco e verifica che il numero di vampiri, licantropi e mutaforma sia accettabile secondo alcuni criteri di giocabilità (i dettagli non interessano). Se la verifica non va a buon fine l'attività termina producendo una schermata di errore nell'interfaccia (i dettagli non interessano). Se invece va a buon fine, concorrentemente si eseguono le due sottoattività di *gioco* e di *verifica* descritte sotto. Una volta che tali sottoattività sono state completate, si preparano i dati per il livello successivo (i dettagli non interessano) e si termina.

L'attività di *gioco* del livello è costituita da una sequenza di passi di interazione tra personaggi. In particolare, inizialmente *(i)* tutti i personaggi sono collocati in modo random nei punti di interesse della mappa, *(ii)* i personaggi hanno un valore di energia pari al valore di accesso al livello, *(iii)* i mutaforma non hanno la forma di altri personaggi. Ad ogni passo tutti i personaggi scelgono un'azione tra quelle a loro disponibili, si definiscono gli scontri, si memorizza un "record" nel log che contiene l'intero stato del gioco (energia dei singoli personaggi, punto di interesse occupato, per i mutaforma i personaggi di cui hanno la forma, ecc.). La sequenza di passi ha temine quando il 10% dei personaggi ha un valore di energia uguale o superiore al valore di energia di uscita del livello.

La sottoattività di *verifica* calcola alcune statistiche nel log prodotto dall'attività di *gioco* man mano che questo viene aggiornato, producendo una stream di dati verso l'interfaccia (i dettagli non interessano). I dettagli delle varie statistiche prodotte non interessano, eccetto che il calcolo del valore atteso di vittorie/sconfitte negli scontri di ciascun personaggio, da mantenere aggiornato con il log prodotto.

Analisi. Basandosi sui requisiti riportati sopra, effettuare l'analisi producendo lo schema concettuale in un formalismo standard per la rappresentazione concettuale, quale ad esempio UML, comprensivo del diagramma delle classi (inclusi vincoli non esprimibili), diagramma degli stati e delle transizioni per i vari tipi di personaggi, diagramma delle attività, con eventuale specifica testuale dei dettagli non rappresentabili direttamente nei diagrammi, motivando, qualora ce ne fosse bisogno, le scelte di progetto.

Progetto e realizzazione. Scegliere le tecnologie di realizzazione discutendone la scelta e illustrando i prodotti di tale fase, incluse le strutture dati scelte, gli aspetti algoritmici più pregnanti e la comunicazione con l'interfaccia (l'interfaccia stessa non è di interesse), motivando le scelte di progetto.

Roma, 20 marzo 2018

Autorità Garante
della Concorrenza e del Mercato
Autorità Nazionale Anticorruzione

Concorso pubblico, per titoli ed esami, a 2 posti nella qualifica di funzionario in prova, uno nel ruolo della carriera direttiva dell'Autorità Garante della Concorrenza e del Mercato (VI° livello della scala stipendiale) e uno nel ruolo dell'Autorità Nazionale Anticorruzione (Categoria A, parametro retributivo F1), per lo svolgimento di attività di indagine, progettazione, sviluppo e di *reverse engineering* di software, algoritmi e data base. (G.U. - IV^ Serie speciale - Concorsi ed esami, n. 56 del 25/7/2017).

Prova tecnico-pratica scritta del 20 marzo 2018

TRACCIA n° 2

Requisiti. L'applicazione da progettare riguarda il sistema di gestione delle partite Laser-Tag. Una partita ha un codice identificativo ed è costituita da diversi giocatori. I giocatori sono divisi in squadre ciascuna delle quali è contraddistinta da un colore (visibile a tutti i giocatori). Ogni partita deve avere almeno 2 squadre. Nel caso le squadre della partita siano almeno 3, le squadre possono allearsi tra loro.
Ogni squadra deve avere almeno 5 giocatori, tra cui uno che ne è il capitano (quest'ultimo è caratterizzato da una divisa diversa visibile a tutti i giocatori). I giocatori sono caratterizzati da un identificatore, da un nome, da un intero compreso tra 0 e 99 che rappresenta la capacità di schivare colpi, da un secondo intero anche esso compreso tra 0 e 99 che rappresenta la mira del giocatore stesso, e da un terzo intero che rappresenta il numero di colpi ricevuti durante la partita. Inoltre i giocatori hanno un flag che denota se sono attualmente visibili o invisibili.

Ogni *giocatore* può eseguire le seguenti azioni.

- *Cerca*, che può essere eseguita quando il giocatore è invisibile. L'azione *Cerca* lo rende visibile lo pone alla ricerca di un qualsiasi avversario, cioè un giocatore visibile di un'altra squadra non alleata, fino a che non ne abbia *trovato* uno o non abbia ricevuto un *colpo* da un giocatore avversario. Durante l'esecuzione di *Cerca* il giocatore è necessariamente visibile. Si noti che *trovato* e *colpo* sono *eventi esogeni*, ciò eventi che non sono sotto il controllo del giocatore ma decisi randomicamente.

- *Spara*, che può essere eseguita solo quando il giocatore è visibile ed ha appena trovato un avversario oppure è stato appena colpito da un avversario. L'azione *Spara* ha come effetto lanciare un *colpo* all'avversario. Quando riceve un colpo, il giocatore incrementa o meno il numero di colpi subiti secondo una funzione booleana (che si assume data) che calcola un valore random sulla base della mira del giocatore che ha sparato e della capacità di schivare del giocatore che subisce il colpo.

- *Nasconditi*, che può essere eseguita solo se il giocatore è stato appena colpito e lo rende invisibile.

In aggiunta, ogni *capitano* può eseguire le due azioni *ChiediAlleanza* e *AccettaAlleanza*. La prima deve essere rivolta al capitano della squadra una volta *trovato,* mentre la seconda può essere eseguita solo se ad un capitano è stata appena chiesta un'alleanza ed ha come effetto di far diventare alleate le due squadre del capitano proponente e del capitano che accetta la proposta. Si noti che queste due azioni sono disponibili solo se le squadre che partecipano al gioco sono almeno 3.

Siamo interessati alla seguente attività principale. L'attività prende in input una partita e concorrentemente esegue le due sottoattività di *gioco* e di *verifica* descritte nel seguito. Una volta che tali sottoattività sono state completate, si preparano i dati necessari per stabilire la o le squadre vincitrici (i dettagli non interessano) e si termina.

L'attività di *gioco* del livello è costituita da una sequenza di passi di interazione tra i giocatori. In particolare, inizialmente tutti i giocatori sono inizializzati con valori della capacità di schivare colpi e di mira random, i colpi subiti vengono posti a 0, ed eventuali alleanze vengono eliminate. Ad ogni passo tutti i personaggi scelgono un'azione tra quelle a loro disponibili oppure continuano l'esecuzione dell'azione scelta precedentemente, si ottengono gli eventi esogeni, si aggiornano i numeri di colpi subiti, e si memorizza un "record" nel log che contiene l'intero stato del gioco alla fine del passo di esecuzione. La sequenza di passi ha termine quando il 50% dei giocatori ha subito almeno 10 colpi.

La sottoattività di *verifica* calcola alcune statistiche nel log prodotto dall'attività di gioco man mano che questo viene aggiornato producendo una stream di dati verso l'interfaccia (i dettagli non interessano). I dettagli delle varie statistiche prodotte non interessano, eccetto che il calcolo del valore atteso di proposte di alleanze e alleanze ottenute tra le squadre, da mantenere aggiornato con il log prodotto.

Analisi. Basandosi sui requisiti riportati sopra, effettuare l'analisi producendo lo schema concettuale in un formalismo standard per la rappresentazione concettuale, quale ad esempio UML, comprensivo del diagramma delle classi (inclusi vincoli non esprimibili), diagramma degli stati e delle transizioni per i vari tipi di giocatori, diagramma delle attività, con eventuale specifica testuale dei dettagli non rappresentabili direttamente nei diagrammi, motivando, qualora ce ne fosse bisogno, le scelte di progetto.

Progetto e realizzazione. Scegliere le tecnologie di realizzazione discutendone la scelta, e illustrando i prodotti di tale fase, incluse le strutture dati scelte e gli aspetti algoritmici più pregnanti (interfaccia stessa non è di interesse), motivando le scelte di progetto.

Roma, 20 marzo 2018

Autorità Garante
della Concorrenza e del Mercato
Autorità Nazionale Anticorruzione

Concorso pubblico, per titoli ed esami, a 2 posti nella qualifica di funzionario in prova, uno nel ruolo della carriera direttiva (VI° livello della scala stipendiale) dell'Autorità Garante della Concorrenza e del Mercato e uno nel ruolo dell'Autorità Nazionale Anticorruzione(Categoria A, parametro retributivo F1) per lo svolgimento di attività di indagine, progettazione, sviluppo e di *reverse engineering* di software, algoritmi e data base. (G.U. - IV^ Serie speciale - Concorsi ed esami, n. 56 del 25/7/2017).

Prova tecnico-pratica scritta del 20 marzo 2018

TRACCIA n° 3

Requisiti. L'applicazione da progettare riguarda una parte di un sistema robotico per l'esplorazione automatizzata di ambienti antartici. Una esplorazione ha un identificatore, una descrizione, un'area da esplorare e un insieme di basi coinvolte. L'area da esplorare è divisa in quadranti adiacenti tra loro, ciascuna con delle coordinate geografiche. Delle basi interessano le coordinate geografiche e una descrizione (una stringa). Una base ha un insieme di dispositivi robotici, ciascuno con un identificatore e con le coordinate geografiche in cui è situato (inizialmente quelle della base). Ogni dispositivo robotico appartiene esattamente a una base. I dispositivi robotici sono suddivisi in rover e droni. Ogni rover ha a bordo uno o più droni. Ogni drone è a bordo di esattamente un rover. Dei rover interessa il raggio di azione dalla base. Dei droni interessa il raggio di azione dal rover su cui sono a bordo.

Il comportamento dei rover e dei droni è il seguente. Un rover è inizialmente alla base. Quando riceve un comando dalla propria base di partire in missione con payload costituito dalle coordinate da raggiungere e codice dell'analisi da effettuare, (1) se la distanza è inferiore al suo raggio di azione, si reca presso le coordinate desiderate e inizia a svolgere l'analisi richiesta secondo una procedura prefissata (i dettagli non interessano); (2) se la distanza è superiore al proprio raggio di azione, ma inferiore alla somma del suo raggio di azione e del raggio di azione del suo drone attualmente a bordo che può andare più lontano, allora si reca alle coordinate a distanza massima dalla base lungo la retta che passa per le coordinate desiderate e per le coordinate della base e fa partire uno dei droni disponibili il cui raggio di azione è superiore alla distanza dal punto corrente alle coordinate desiderate (si assuma di avere già disponibili la funzione per la selezione del drone e la funzione di calcolo delle coordinate di destinazione del rover); (3) se la distanza è superiore, invia alla base comunicazione dell'impossibilità di eseguire la missione senza spostarsi. Se una volta in missione riceve il comando di rientrare alla base, interrompe la missione e nel caso (1) torna alla base, mentre nel caso (2) se ha già fatto partire il drone, manda un comando di rientro al drone scelto restando in attesa del segnale di rientro da parte del drone e, quando riceve tale segnale, torna alla base, altrimenti torna direttamente alla base.

Invece, ogni drone è inizialmente presso il proprio rover. Quando un drone riceve un comando di esplorazione dal proprio rover con payload le coordinate da raggiungere e il codice dell'analisi da effettuare, si alza in volo, le raggiunge e inizia a svolgere l'analisi richiesta secondo una procedura

prefissata (i dettagli non interessano). Quando riceve il comando di rientrare, interrompe la missione e ritorna al rover segnalando il suo arrivo.

Siamo interessati alla seguente attività principale. L'attività prende in input un'esplorazione e verifica che le basi coinvolte i loro rover e droni siano capaci di coprire l'area (i dettagli non interessano). Se la verifica non va a buon fine, l'attività termina. Altrimenti viene stabilito un piano di esplorazione dove ogni base manda in missione e richiama (anche ripetutamente) i propri rover secondo sottopiani opportunamente stabiliti da un protocollo calcolato (i cui dettagli non interessano). Dopodiché concorrentemente si eseguono le due sottoattività di esplorazione vera e propria e di verifica descritte sotto. Una volta che tali sottoattività sono state completate, l'attività principale termina.

L'attività di *esplorazione* è costituita dall'esecuzione da parte dei rover e dei loro droni del piano di esplorazione. Inoltre, secondo un intervallo temporale prestabilito, si memorizza un "record" nel log che contiene l'intero stato dell'esplorazione: quali rover e droni sono in missione, quale posizione hanno, cosa stanno facendo (e.g., recandosi alle coordinate desiderate, analizzando, tornado alla base o al rover, ecc.).

La sottoattività di *verifica* calcola alcune statistiche nel log prodotto dall'esplorazione man mano che questo viene aggiornato, producendo una stream di dati verso l'esterno dell'applicazione (i dettagli non interessano). I dettagli delle varie statistiche prodotte non interessano, eccetto che il calcolo del valore atteso di droni in missione per ciascun rover di ciascuna base, da mantenere aggiornato con il log prodotto.

Analisi. Basandosi sui requisiti riportati sopra, effettuare l'analisi producendo lo schema concettuale in un formalismo standard per la rappresentazione concettuale, quale ad esempio UML, comprensivo del diagramma delle classi (inclusi vincoli non esprimibili), diagramma degli stati e delle transizioni per i vari tipi di sistemi robotici, diagramma delle attività, con eventuale specifica testuale dei dettagli non rappresentabili direttamente nei diagrammi, motivando, qualora ce ne fosse bisogno, le scelte di progetto.

Progetto e realizzazione. Scegliere le tecnologie di realizzazione discutendone la scelta, e illustrando i prodotti di tale fase, incluse le strutture dati scelte e gli aspetti algoritmici più pregnanti, motivando le scelte di progetto.

Roma, 20 marzo 2018