Diagramma UML carte avventure

Elenco entita

Il macro-aspetto delle carte avventure e' composto dai seguenti componenti:

Interfacce che definiscono dei metodi (default) che verranno usati nelle varie classi rappresentanti le carte avventura. Queste interfacce sono 7 e sono: "HaSpostamento" (1), "HaRequisito" (2), "HaGuadagnoCrediti"(3), "HaPerditaPerColpi" (4), "CalcoloMinimoAbitanti" (5), "HaPerditaSegnalini" (6), "HaGuadagnoMercie" (7).

Una classe astratta di nome "Carta" che raccoglie tutti gli aspetti comuni a tutte le altre carte.

Una classe per ogni carta del gioco, che eredita' dalla classe "Carta" ed implementa le interfacce che gli servono.

Una classe ausiliaria di nome "Colpo" che rappresenta un colpo. Il colpo ha un'orientamento e puo' essere grande o piccolo, il tipo del colpo viene definito nella carta stessa tramite un attributo.

Una enumerazione denominata "TipoElemento" che puo' assumere i valori Meteorite, Cannonata, Merce, PotenzaDiFuoco, PotenzaMotrice.

Spiegazione entita

Tutte le carte hanno i metodi risolvi(giocatore) e isRisolta(void). Questi metodi devono essere chiamati dall'entita' gestore. All'interno del metodo avvia() del gestore ci sara' la seguente porzione di codice per la gestione della risoluzione di una carta:

```
while(mazzo.getDim() > 0){
        turno = 4;
        cartaAttuale = mazzo.getCarta();
        while(!cartaAttuale.isRisolta()){
                cartaAttuale.risolvi(Giocatori[turno]);
                turno--;
        }
}
```

Giocatori e' un array che contiene tutti i giocatori, turno un intero che parte da 4, rappresentante il giocatore in testa nella plancia volo, e scende mano mano. Il metodo risolvi della carta attuale si occupa di gestire la giocata del giocatore indicato. Per esempio, nel caso della carta contrabbandieri, appena un giocatore la risolve (cioe' sconfigge i contrabbandieri), il metodo risolvi porra' a true l'attributo risolto, e la carta sara' quindi stata risolta. Se invece non la risolve, l'attributo risolvi rimane a false, turno viene decrementato e tocchera' al giocatore immediatamente successivo risolvere eventualmente la carta, o pagarne le conseguenze.

Le interfacce definiscono dei metodi ausiliari default che vengono usati all'interno del metodo risolvi di ogni carta. Attenzione, il gestore usa unicamente i metodi isRisolta e risolvi.

Gli attributi comuni a tutte le carte sono nomeCarta, livello e risolta. Ogni carta ha poi ovviamente i suoi attributi specifici. Gli attributi sono costanti, di conseguenza il costruttore delle carte non ha argomenti, il che semplifica di molto la fase di istanziazione delle carte.

Le carte vengono istanziate ad inizio partita in base al tipo della partita che si sceglie di fare (volo di prova, livello 1 o livello 2). Ci vuole quindi una classe all'interno nel macro-aspetto assemblaggio che prende come argomento il tipo della partita scelta e istanzia le carte in base a quest'ultima, restituendo il mazzo principale della partita, composto da un array con tutte le carte istanziate.

I metodi delle varie interfacce, per poter applicare la carta al giocatore, dovranno agire sulla plancia volo e sulla plancia nave del giocatore. Di conseguenza dovranno utilizzare dei metodi definiti nella plancia volo e nella plancia nave.

Le interfacce hanno i seguenti ruoli (autoesplicativi dal loro nome):

1) HaSpostamento

Definisce un metodo che fa spostare il giocatore indicato di x posizioni. Avanti o indietro in base al segno di x.

2) HaRequisito

Definisce un metodo che restituisce true o false in base alla soddisfacibilita' o no di un determinato requisito per il giocatore indicato. I requisiti possono essere di potenza di fuoco, potenza motrice o abitanti/alieni.

3) HaGuadagnoCrediti

Definisce un metodo che aggiunge i crediti indicati al giocatore indicato.

4) HaPerditaPerColpi

Definisce un metodo che gestisce i colpi che un giocatore indicato subisce.

5) HaCalcoloMinimoAbitanti

Definisce un metodo che restituisce il giocatore con il minor numero di abitanti. Serve per carte come zona di guerra o sabotaggio.

6) HaPerditaSegnalini

Definisce un metodo che fa perdere al giocatore indicato i segnalini indicati. I segnalini possono essere abitanti o merce.

7) HaGuadagnoMerce

Definisce un metodo che fa guadagnare al giocatore indicato la merce indicata.

Diagramma UML

Il diagramma ha la seguente struttura:

Nello strato basso ci sono la classe astratta “carta”, la classe ausiliaria “Colpi” e l’enumerazione “Tipo elemento”. Nello strato intermedio ci sono tutte le classi rappresentanti le carte e nello strato altro tutte le interfacce.