ALMA MATER STUDIORUM - UNIVERSITÀ DI BOLOGNA

FACOLTA' DI INGEGNERIA

CORSO DI LAUREA IN INGEGNERIA INFORMATICA

PROGETTO DI INGEGNERIA DEL SOFTWARE

a.a. 2011/2012

professor GIUSEPPE BELLAVIA

di:

LORENZO DONINI

ALESSANDRO PIOVANI

ELISA TARTARINI

INDICE

RACCOLTA REQUISITI

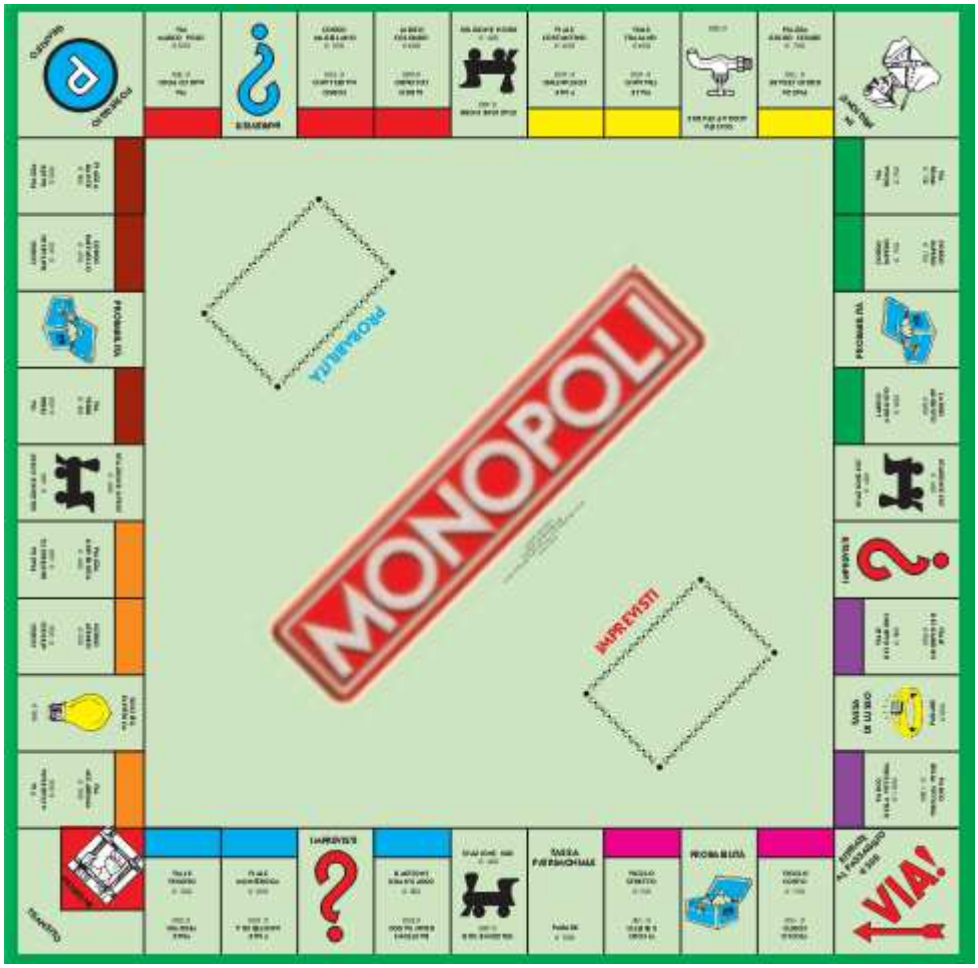
SPECIFICHE

Si chiede di realizzare un videogame in 2D dell'edizione classica del gioco di società Monopoli Parker Brothers.

Le specifiche di progetto, pertanto, sono le medesime esposte nel regolamento del gioco stesso.

Inoltre:

- La moneta virtuale da utilizzare è l'euro.
- Sono previsti soltanto giocatori umani e non giocatori governati da intelligenza artificiale.
- I giocatori interagiscono con il videogame tramite appositi pulsanti grafici.
- Il videogame deve essere interamente compatibile con piattaforma Microsoft Windows (Windows XP, Windows Vista, Windows 7).
- La creazione di una nuova partita non deve impiegare più di un secondo e tutti i comandi di gioco devono essere eseguiti in modo fluido e veloce



MONOPOLI
Regolamento

LO SPIRITO DEL GIOCO

Lo scopo del gioco è di trarre profitto, affittando, comprando e vendendo le proprietà che si trovano lungo il percorso della plancia di gioco, sino a diventare il giocatore più ricco e, possibilmente, il MONOPOLISTA.
Partendo dal VIA!, ogni giocatore a turno muove il proprio segnalino lungo il percorso, in base al punteggio ottenuto con i dadi.
Quando il segnalino si ferma su una proprietà libera, il giocatore può acquistarla dalla Banca, pagando il prezzo indicato sulla casella e riportato anche sul contratto, altrimenti la proprietà viene subito messa all'asta e ceduta al miglior offerente.
Chi possiede una proprietà ne gode la rendita, la quale è costituita dall'affitto che ogni altro giocatore, fermandosi sulla proprietà, è tenuto a pagargli.
Le rendite o gli affitti aumentano in ragione delle case e degli alberghi che vengono eretti sul terreno, per cui è consigliabile fabbricare il più possibile sull'area posseduta.
Per avere maggiore denaro liquido e intensificare le costruzioni, si possono ottenere dalla Banca ipoteche sulle proprietà.
Quando capita su una casella segnata PROBABILITÀ o IMPREVISTI, il giocatore di turno deve prendere un cartoncino dal relativo mazzo e seguire fedelmente le indicazioni in esso contenute.
Nel corso del gioco si può anche finire in prigione e uscire o per condono o mediante il pagamento di una somma.

REGOLAMENTO

Da due a sei giocatori (anche sette, quando chi fa da banchiere non partecipa al gioco).
La PLANCIA di gioco è suddivisa in 40 caselle che rappresentano terreni, società, stazioni, tasse, probabilità, imprevisti ecc. Sopra queste caselle si muovono i segnalini che ogni giocatore sceglie all'inizio del gioco.
Per determinare gli spostamenti, si utilizzano due DADI, mentre le costruzioni sono rappresentate dalle CASE verdi e dagli ALBERGHI rossi.
Per fissare le rendite e le ipoteche, vi sono tanti contratti quanti sono i terreni, le società e le stazioni.

PREPARAZIONE

Scegliete un segnalino ciascuno e collocateli tutti sulla casella VIA!.
Decidete quale giocatore farà il Banchiere, compito che non comporta nessun vantaggio particolare e che può essere ricoperto anche da qualcuno che non partecipa direttamente al gioco.
In base al numero dei partecipanti, il Banchiere distribuisce ad ogni giocatore un capitale iniziale e una serie di contratti. I contratti devono essere distribuiti casualmente, dopo averli accuratamente mischiati.

Numero giocatori	Denaro Iniziale	Numero contratti
2	€ 8.750,00	7
3	€ 7.500,00	6
4	€ 6.250,00	5
5	€ 5.000,00	4
6	€ 3.750,00	3

Prima di iniziare la partita, i giocatori devono pagare alla Banca il valore di acquisto dei contratti che hanno ricevuto.

INIZIO DEL GIOCO

Tutti i giocatori, dopo aver posizionato i propri segnalini sulla casella del VIA!, lanciano i due dadi e chi totalizza il punteggio più alto comincia il gioco.
Il giocatore di turno lancia i dadi e muove il proprio segnalino di un numero di caselle pari al punteggio ottenuto, procedendo nel senso indicato dalla freccia che si trova sulla casella del VIA!, esegue le eventuali operazioni collegate alla casella di arrivo e, poi, passa i dadi a chi è seduto alla sua sinistra, che diventa il nuovo giocatore di turno.
Due o più segnalini possono trovarsi contemporaneamente sulla stessa casella, senza che questo comporti alcun mutamento del normale svolgimento del gioco.
A seconda della casella sulla quale si ferma il segnalino, al giocatore si presentano queste diverse possibilità:

- diventare proprietario del terreno (se è ancora "libero")
- pagare un affitto al proprietario del terreno
- pagare una tassa
- pescare un cartoncino PROBABILITÀ oppure IMPREVISTI (a seconda del simbolo indicato sulla casella)
- andare in prigione

Dalla casella PRIGIONE si può anche transitare liberamente, mentre la casella POSTEGGIO GRATUITO non comporta alcun tipo di azione da parte del giocatore di turno.
Quando un giocatore ottiene con i dadi un punteggio doppio (ad esempio 1-1), procede con il segnalino come di consueto, ma al termine del turno, dopo aver sottostato a quanto previsto dalla casella raggiunta, lancia nuovamente i dadi per un altro turno.
Un giocatore che ottiene tre volte di seguito un punteggio doppio, deve andare direttamente in prigione e senza passare dal VIA!.

Indennità

Ogni volta che un giocatore si ferma o transita dal VIA!, riceve dalla Banca € 500.

Fermata su proprietà libera

Quando un giocatore capita su una casella rappresentante una proprietà che non è ancora stata aggiudicata, ha diritto ad acquistarla pagando alla Banca il prezzo indicato. Il giocatore non è obbligato all'acquisto, ma in questo caso la proprietà viene immediatamente messa all'asta fra tutti i giocatori (compreso chi non ha voluto acquistarla al prezzo pieno). Il prezzo di partenza per l'asta è di € 5 per qualsiasi proprietà.

Chi entra in possesso di una proprietà riceve dalla Banca il relativo contratto che ne prova l'acquisto e che deve essere posto sul tavolo in modo ben visibile a tutti, con la parte colorata rivolta verso l'alto.

Fermata su proprietà aggiudicata

Quando un giocatore si ferma su una proprietà di un avversario, deve pagare al proprietario l'affitto indicato sul contratto. Se sul terreno sono stati costruiti degli immobili (case e/o alberghi), l'affitto aumenta come indicato sul contratto stesso.

Se la proprietà è stata in precedenza ipotecata (e in questo caso il contratto deve essere girato dal lato indicante l'ipoteca), il proprietario non può richiedere alcun affitto.

Vantaggi per i proprietari

Aggiudicarsi la proprietà di un gruppo completo di terreni dello stesso colore è molto vantaggioso perché il proprietario può esigere un affitto doppio (se ancora non vi ha costruito case o alberghi). Inoltre, chi possiede il monopolio dei terreni di uno stesso colore può iniziare a costruirvi case ed alberghi, potendo esigere affitti sempre più elevati.

Anche possedere entrambe le società o più di una stazione comporta vantaggi che sono indicati sui relativi contratti.

Probabilità o Imprevisti

Il giocatore che si ferma su una di queste caselle, deve pescare il primo cartoncino relativo alla casella raggiunta e, dopo aver eseguito le istruzioni indicate, rimetterlo in fondo al mazzo. Solo i cartoncini che consentono di USCIRE GRATIS DI PRIGIONE possono essere trattenuti fino a che non vengono utilizzati (e rimessi in fondo al mazzo).

Tasse, Posteggio gratuito e Transito

Chi si ferma sulla casella delle Tasse è tenuto a pagare alla Banca la cifra indicata. Chi si ferma sulla casella POSTEGGIO GRATUITO o TRANSITO (su questa esiste anche la prigione) vi rimane senza subire alcuna conseguenza.

Banca e Banchiere

Il compito di fare da banchiere viene affidato a chi abbia, possibilmente, le migliori doti di banditore d'asta. Se, come solitamente avviene, il Banchiere è contemporaneamente un giocatore, egli deve tenere il denaro della Banca separato dai suoi fondi personali, essendo le operazioni di Banca del tutto estranee alle vicende dei giocatori. La Banca paga indennità e premi, incassa le tasse e i pagamenti in genere, vende le proprietà, consegna ai giocatori i contratti, le case e gli alberghi e concede le ipoteche. I giocatori, in qualsiasi momento del gioco, possono rivendere le loro case

alla Banca a metà del prezzo indicato sul contratto di proprietà del terreno sul quale erano state costruite.

Prigione

Un giocatore finisce in prigione se:

1) Il segnalino termina sulla casella IN PRIGIONE!
2) Pesca un cartoncino PROBABILITÀ o IMPREVISTI sul quale è scritto di andare in prigione
3) Ottiene per tre volte di seguito un punteggio doppio con i dadi.

La PRIGIONE è collocata nella stessa casella del TRANSITO. Chi è condannato alla Prigione ci va direttamente e senza mai passare dal VIA!, il che significa che non riceve € 500 di indennità. Se un giocatore capita sulla casella della Prigione in seguito ad un normale lancio di dadi, vi resta soltanto come visitatore in transito e, al proprio turno, prosegue.

Chi si trova in PRIGIONE può uscirne in uno dei seguenti modi:

1. mediante il pagamento di € 125 al suo prossimo turno, prima di lanciare i dadi
2. ottenendo un punteggio doppio con i dadi senza dover pagare nulla. In questo caso deve muovere il segnalino di un numero di caselle pari al punteggio ottenuto
3. utilizzando uno dei cartoncini sui quali è scritto USCITE GRATIS DI PRIGIONE.

Ad ogni turno trascorso in Prigione, il giocatore deve comunque lanciare i dadi e, dopo tre turni, è costretto a pagare € 125 di multa e uscire, utilizzando per il movimento l'ultimo punteggio ottenuto.

Mentre si è in prigione, si può continuare ad acquistare case ed alberghi e partecipare alle eventuali aste.

Case ed alberghi

La vendita delle case e degli alberghi è riservata alla Banca. Un giocatore può costruire case e/o alberghi solo quando possiede TUTTI i terreni di uno stesso gruppo, cioè dello stesso colore. Case ed alberghi possono essere acquistati in qualsiasi momento del gioco, ma non dopo aver visto il punteggio ottenuto con i dadi da un avversario e prima che questi abbia mosso il suo segnalino. La costruzione delle case deve avvenire in modo proporzionato sui terreni dello stesso colore, in modo che su ogni terreno vi sia lo stesso numero di case o, al massimo, una sola casa in più rispetto agli altri.

Su ogni terreno vi possono essere al massimo 4 case. Il prezzo di ogni casa dipende dal terreno sul quale si vuole edificare ed è indicato sul relativo contratto.

Gli alberghi si costruiscono restituendo alla Banca le 4 case già esistenti e pagando in più il prezzo indicato sul contratto di proprietà.

Il valore di un albergo corrisponde a quello di 5 case.

Su ogni terreno non è possibile costruire più di un albergo e non si possono costruire alberghi a meno che non si possano prima costruire 4 case.

Vendita di proprietà

I terreni senza case o alberghi, le stazioni e le società, possono essere oggetto di trattativa fra i giocatori e scambiate o vendute liberamente. Un terreno sul quale sono state edificate case o alberghi non può essere venduto o scambiato se prima non

vengono rivenduti alla Banca, a metà prezzo, le case e/o gli alberghi su di esso precedentemente edificati.

Ipoteche

La Banca è la sola a poter concedere ipoteche. Il valore ipotecario di ogni terreno, società o stazione è segnato sul relativo contratto e corrisponde alla met à del prezzo originario. Quando si vuole riscattare l 'ipoteca esistente su un terreno, occorre pagare alla Banca il valore dell'ipoteca maggiorato del 10% (arrotondando sempre per eccesso). Se si vende un terreno ipotecato ad un altro giocatore, il compratore deve pagare alla Banca il 10% dell'ipoteca. Nello stesso tempo, egli può riscattare l ipoteca, pagandone il prezzo alla Banca. Se, invece, si limita al pagamento del 10% (senza riscattare completamente l'ipoteca), quando vorrà riscattarla dovrà pagare ancora una volta il 10% in più.

Un terreno può essere ipotecato soltanto se tutto il gruppo al quale appartiene è privo di costruzioni. Ove vi siano delle costruzioni, esse vanno vendute alla Banca che le pagherà metà del loro prezzo di acquisto. Su terreni ipotecati non è possibile costruire nè case nè alberghi fino al riscatto dell'ipoteca.

Fallimento

Quando un giocatore deve pagare alla Banca o ad un altro giocatore, una somma superiore a tutto ciò che possiede, comprendendo sia il denaro liquido, sia le case e gli alberghi rivenduti a metà prezzo alla Banca, sia i terreni liberi al prezzo d'ipoteca, allora fallisce. In tal caso tutto ciò che egli possiede passa alla Banca che paga integralmente il creditore e rimette immediatamente le proprietà all'asta, ad esclusione delle case e degli alberghi.

Il giocatore fallito deve ritirarsi dal gioco

Conclusione del gioco

Quando il penultimo giocatore fallisce, l'ultimo rimasto in gioco vince la partita. Questa è la conclusione "normale" del gioco, però è anche possibile giocare una partita più breve. All'inizio del gioco, si stabilisce la durata della partita (es. 90 minuti). Allo scadere del tempo, i giocatori conteranno le proprie banconote aggiungendo i valori dei terreni, delle società, delle stazioni e degli edifici. Le case e gli alberghi devono essere calcolati alla metà del loro valore di acquisto.

VINCE IL PIÙ RICCO.

REQUISITI

Lo studio di fattibilità e l'analisi dei requisiti ha portato ad apportare alcune modifiche alle specifiche iniziali:

- L'ordine in cui i giocatori si susseguono nel giocare i loro turni durante la partita è stabilito dal sistema in modo casuale.
- Sono proibiti gli scambi di proprietà tra i vari giocatori. L'unico modo per entrare in possesso di un terreno di un altro giocatore, è aspettare che quel terreno torni libero, ovvero di proprietà della Banca.
- Il gioco non prevede le carte "uscite gratis di prigione", dunque sono assenti negli imprevisti e nelle probabilità. Gli unici modi per uscire di prigione sono il pagamento della cauzione o un doppio numero con i dadi (al terzo turno di prigione è comunque obbligatorio pagare la cauzione).
- Il gioco non prevede le aste dei terreni.
- I terreni non possono essere ipotecati. Un giocatore ha tuttavia la possibilità di vendere i terreni in sua proprietà alla Banca per un importo pari alla metà del valore originale.
- Gli edifici possono essere eretti sui terreni da un giocatore solamente durante il suo turno e possono essere distribuiti a piacere sui terreni dello stesso gruppo.

Inoltre sono stati introdotti requisiti riguardanti la giocabilità del videogioco:

- Per poter giocare è necessario creare una nuova partita. Non possono essere create più partite contemporaneamente: per creare una nuova partita bisogna aver terminato quella in corso.
- Deve essere possibile creare una partita a tempo, scaduto il quale la partita termini automaticamente. Il tempo massimo deve essere stabilito all'inizio del gioco.
- Durante lo svolgimento di una partita, ogni giocatore può abbandonare la partita in qualsiasi momento durante il suo turno; tutte le sue proprietà torneranno in possesso della banca, e non gli sarà più possibile rientrare in partita.
- La Banca è interamente governata dal software, dunque non da giocatori umani.
- Al termine di una partita il software deve calcolare il valore complessivo dei lotti posseduti da ogni giocatore (e relativi edifici) più il proprio capitale e porre l'elenco risultante in una schermata di statistiche; il giocatore con il valore complessivo maggiore vince la partita.

GLOSSARIO

Le parti evidenziate indicano specifiche presenti inizialmente, ma eliminate durante l'analisi dei requisiti.

TERMINE	SIGNIFICATO E CARATTERISTICHE
“In Prigione”	Casella che compare una sola volta sulla tavola da gioco. Un giocatore che si ferma con il proprio segnalino su questa casella è costretto ad andare in Prigione.
“Posteggio Gratuito”	Casella che compare una sola volta sulla tavola da gioco. Un giocatore che si ferma con il proprio segnalino su questa casella non subisce alcuna conseguenza.
“Tassa di lusso”	Casella di tipo Tassa, caratterizzata da un importo. Compare solo una volta sulla tavola da gioco.
“Tassa patrimoniale”	Casella di tipo Tassa, caratterizzata da un importo. Compare solo una volta sulla tavola da gioco.
“Via!”	Casella della tavola da gioco. Compare una sola volta sulla tavola da gioco. All'inizio di una partita tutti i giocatori pongono il proprio segnalino su questa casella. A ogni successivo passaggio da questa casella (il giocatore deve passarci sopra e non necessariamente sostare), ogni giocatore riceve un'indennità pagata dalla banca.
Acquisto di un terreno	Quando un giocatore finisce con il proprio segnalino su un terreno non posseduto da alcun giocatore, ma in possesso della Banca, egli può decidere di acquistare quel terreno. Qualora decida di acquistarlo, deve pagare l'importo relativo (indicato sul contratto di quello specifico terreno) alla banca. Una volta pagato l'importo, il terreno diventa di proprietà del giocatore, il quale deve da quel momento in poi chiedere l'affitto.
Affitto	Importo che un giocatore deve pagare a un altro giocatore quando capita con il proprio segnalino su un terreno posseduto da un altro giocatore. L'importo da pagare dipende dal numero di terreni dello stesso gruppo del terreno in questione, posseduti dal giocatore che richiede l'affitto. Maggiore è il numero di terreni di uno stesso gruppo posseduti, maggiore sarà l'affitto da richiedere (come indicato specificamente sul contratto di ogni singolo terreno).
Albergo	Edificio che può essere costruito su un terreno normale dopo aver pagato il relativo prezzo di costruzione alla banca. È possibile edificare solamente un albergo per ogni terreno posseduto, sempre rispettando le regole di costruzione degli edifici. Il valore di un albergo è quello di cinque case.

Banca	È un'entità governata interamente dal software, che ha il compito di: - pagare indennità e premi - incassare le multe e le tasse - vendere le proprietà e consegnarne i relativi contratti ai giocatori - comprare le proprietà dai giocatori se questi vogliono rivenderle alla banca - vendere case e alberghi ai giocatori - comprare case e alberghi dai giocatori Tutte queste azioni vengono scatenate automaticamente a fronte di operazioni effettuate dai giocatori.
Capitale	Somma di denaro virtuale associata a ogni giocatore, necessaria per tutti i vari pagamenti: - vendite - acquisti - tasse - affitti
Carta	Viene pescata dai Giocatori, che devono eseguirne le relative istruzioni. Una volta eseguite le istruzioni, la carta non è più valida e viene rimessa in fondo al mazzo.
Casa	Edificio che può essere costruito su un terreno normale dopo aver pagato il relativo prezzo di costruzione alla banca. È possibile edificare fino a un massimo di quattro case per ogni terreno posseduto, sempre rispettando le regole di costruzione degli edifici.
Casella	Sezione della tavola da gioco. Vi si muovono sopra i segnalini dei giocatori. Discrimina il comportamento del giocatore nel suo turno.
Conclusione	Se rimane un solo giocatore in gioco, se è trascorso il tempo massimo stabilito all'inizio della partita, oppure se un Giocatore decide di terminare arbitrariamente la Partita, si entra nella fase conclusiva della partita, in cui i Giocatori non possono più giocare dei turni. Durante questa fase il sistema elabora e presenta il vincitore della partita assieme a una schermata di statistiche. Le statistiche finali tengono conto anche di eventuali Giocatori inattivi.
Contratto	Documento associato ad ogni terreno, che ne contiene le relative informazioni.

Contratto normale	Contratto relativo a un terreno normale, che ne contiene: - nome e gruppo - valore - valore di vendita - tassa affittuaria - prezzo di costruzione degli edifici
Contratto speciale	Contratto relativo a un terreno speciale, che ne contiene: - nome e gruppo - valore - valore di vendita - tassa affittuaria
Edificio	Può essere costruito (acquistandolo dalla banca) sui terreni normali di cui si è proprietari, ma solamente se si possiedono tutti i terreni dello stesso gruppo. Si possono acquistare due tipi di edifici: - le Case - gli Alberghi E' tuttavia possibile costruire su un terreno un albergo solamente se si sono già costruite un totale di quattro case su quel terreno. In tal caso le quattro case vengono automaticamente riconsegnate alla banca e al loro posto viene edificato l'albergo. Tutti i costi di costruzione degli edifici sono specificati sui contratti dei terreni. Ogni giocatore ha la possibilità di vendere un edificio (costruito su un terreno di sua proprietà) alla banca per metà del valore di acquisto. Quando un edificio viene venduto esso viene demolito dal terreno.
Fallimento	Quando un Giocatore non è più in grado di pagare un importo, né con il capitale, né vendendo alla Banca tutti gli edifici e i terreni posseduti, fallisce e diventa inattivo. Qualora il giocatore fallisca non riuscendo a pagare l'affitto a un altro giocatore, questo verrà pagato dalla Banca.
Giocatore	Partecipante alla partita, identificato da un nome e da una pedina. Ha associato un capitale e zero o più contratti. Un giocatore interagisce col sistema giocando dei turni.
Giocatore Inattivo	Quando un Giocatore diventa inattivo, il sistema riconsegna tutti i suoi possedimenti (capitale, terreni, edifici) alla Banca. Un Giocatore inattivo non può più interagire con il sistema, e il suo segnalino viene rimosso dalla tavola da gioco.
Gruppo	Categoria in cui rientra ogni terreno. Il gruppo dei terreni normali è dato da un colore.
Imprevisti	Tipo di casella che compare più volte sulla tavola da gioco. Un giocatore che si ferma con il proprio segnalino su una di queste caselle deve pescare una Carta dal mazzo degli imprevisti.

Indennità	Importo prefissato che un giocatore riceve dalla Banca ogni volta che transita dalla casella “Via!”
Ipoteca	Un Giocatore può decidere di ipotecare un terreno sul quale non siano costruiti edifici. Solo la banca può concedere ipoteche. Il valore ipotecario di ogni terreno, società o stazione è segnato sul relativo contratto e corrisponde alla metà del prezzo originario. Quando si vuole riscattare l’ipoteca esistente su un terreno, occorre pagare alla Banca il valore dell’ipoteca maggiorato del 10%.
Lotto = Terreno	
Partita	Sessione di gioco in cui un numero di giocatori (da 2 a 6) si confrontano al gioco Monopoli. E' composta da : - una preparazione - uno svolgimento del gioco, durante il quale i giocatori devono a rotazione giocare dei turni - una conclusione
Preparazione	Fase della partita durante la quale vengono assegnati ad ogni giocatore un nome, un segnalino, un capitale di partenza e dei contratti. Gli utenti stabiliscono inoltre una durata massima della partita.
Prezzo di costruzione	Costo di un edificio che un giocatore deve pagare alla banca ogni volta che vuole costruire un edificio. L'importo che un giocatore riceve quando decide di restituire un edificio alla banca (togliendolo dal proprio terreno) è pari alla metà del prezzo di costruzione dell'edificio stesso.

Prigione	Casella che compare una sola volta sulla tavola da gioco. Un giocatore può finire in prigione se: 1) pesca una carta che porti le istruzioni di andare in prigione 2) finisce con il proprio segnalino sulla casella “In Prigione” 3) tira i dadi doppi per tre volte di seguito Un giocatore che finisce in prigione non passa dalla casella “Via” e non può muovere il proprio segnalino se non dopo essere uscito di prigione. Se invece un giocatore si ferma con il proprio segnalino su questa casella dopo un normale tiro di dadi non subisce alcuna conseguenza. Un giocatore può uscire di prigione se: 1) paga alla Banca una cauzione prima del tiro dei dadi 2) effettua un tiro di dadi doppio 3) utilizza una Carta “Uscita gratis di prigione” Un Giocatore che si trova in Prigione da tre turni e non ha effettuato un tiro di dadi doppio, è obbligato a pagare la cauzione. Appena uscito di prigione, il Giocatore muove il proprio segnalino secondo il risultato dei dadi precedentemente lanciati.
Probabilità	Tipo di casella che compare più volte sulla tavola da gioco. Un giocatore che si ferma con il proprio segnalino su una di queste caselle deve pescare una Carta dal mazzo delle probabilità.
Proprietà = Terreno	È una casella che può essere posseduta da un giocatore, pertanto è un terreno (edificabile o non).
Ritiro di un Giocatore	Un giocatore può decidere, in qualsiasi momento durante il proprio turno, di ritirarsi dalla partita, diventando così inattivo.
Segnalino	Immagine simbolica colorata che rappresenta un determinato giocatore sulla tavola da gioco.
Stazione Ferroviaria	Stazione per servizi di trasporti, è un tipo di terreno speciale, di cui esistono in totale quattro caselle sulla tavola da gioco.
Tassa	Importo che un Giocatore deve pagare alla Banca dopo essere capitato con il proprio segnalino su una casella di tipo Tassa (l'ammontare da pagare è indicato sulla casella stessa), oppure dopo aver pescato una carta dal mazzo delle Probabilità o degli Imprevisti le cui istruzioni impongono il pagamento di una Tassa alla Banca. Esistono due caselle di tipo Tassa: “Tassa patrimoniale” e “Tassa di lusso”.
Tassa affittuaria = Affitto	
Tavola	Tavola da gioco con 40 caselle indicanti terreni, stazioni per servizi di trasporti, imprese di pubblica utilità, tasse, fermate, premi e penalità.

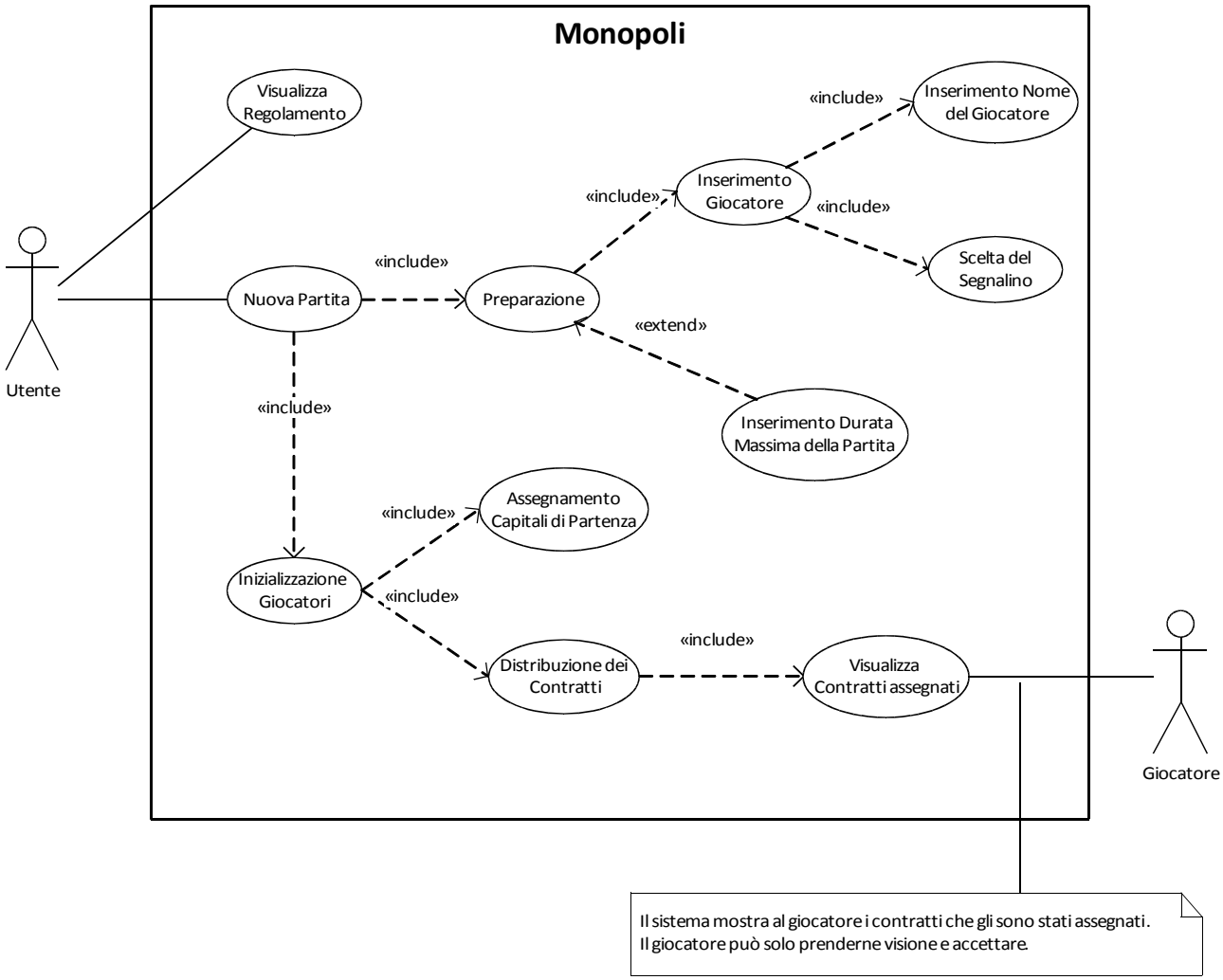
Terreno	Tipo di casella che compare una sola volta sulla tavola da gioco e che può essere acquistata e venduta dai giocatori. Può essere un terreno normale o un terreno speciale.
Terreno normale	Tipo di terreno caratterizzato da un gruppo, sul quale possono essere costruiti e demoliti degli edifici.
Terreno speciale	Tipo di terreno che può essere di due tipologie: stazione ferroviaria, impresa di pubblica utilità. Su nessuno dei terreni speciali si possono costruire degli edifici.
Tipologia = Gruppo	
Tiro di dadi doppio	Particolare tiro di dadi in cui il risultato dei due dadi lanciati è uguale. In seguito a un tiro di dadi doppio un giocatore, alla fine del suo turno, deve effettuare un nuovo turno (rilanciando quindi i dadi).
Turno	Fase ripetitiva del gioco durante la quale un giocatore deve tirare i dadi e muovere il proprio segnalino di un numero di caselle pari alla somma dei dadi. In base alla Casella su cui finisce il segnalino, il Giocatore dovrà e/o potrà effettuare determinate operazioni: - acquistare un terreno - vendere un terreno - costruire un edificio - demolire un edificio - pagare una tassa - pescare una carta - pagare l'affitto Se un Giocatore si trova in Prigione, valgono tuttavia le regole relative alla Prigione. Un Giocatore, durante qualsiasi momento del suo turno, può: - ritirarsi - terminare la partita, entrando così nella fase conclusiva della Partita stessa.
Uscita gratis di prigione	Azione che può essere eseguita in qualsiasi momento se si è pescata una carta dal mazzo delle Probabilità o degli Imprevisti con la seguente descrizione: <<Uscite gratis di prigione>>. Un giocatore può tenere questa carta e utilizzarla in qualsiasi momento oppure venderla a un altro giocatore. Una volta utilizzata, la carta deve essere riposta in fondo al mazzo relativo.
Valore	Importo che un giocatore paga alla banca quando acquista uno specifico terreno.
Valore di Vendita	Importo che la banca paga a un giocatore quando questo le vende uno specifico terreno. Si possono vendere sia terreni normali che terreni speciali (ovvero stazioni ed imprese).

Vendita di un Terreno	Un giocatore, durante il suo turno, può vendere uno o addirittura più terreni di cui è in possesso alla Banca, la quale pagherà un importo pari alla metà del valore del terreno al giocatore. In caso sul terreno vi siano costruiti edifici, verranno automaticamente venduti anche questi alla Banca.
Vincitore della Partita	Durante la Conclusione, il sistema stabilisce il vincitore della partita calcolando il valore complessivo dei terreni posseduti da ogni giocatore (e relativi edifici) più il proprio capitale. Il Giocatore che risulta più ricco è il vincitore.

ANALISI DEI REQUISITI FUNZIONALI – MODELLO COMPORTAMENTALE

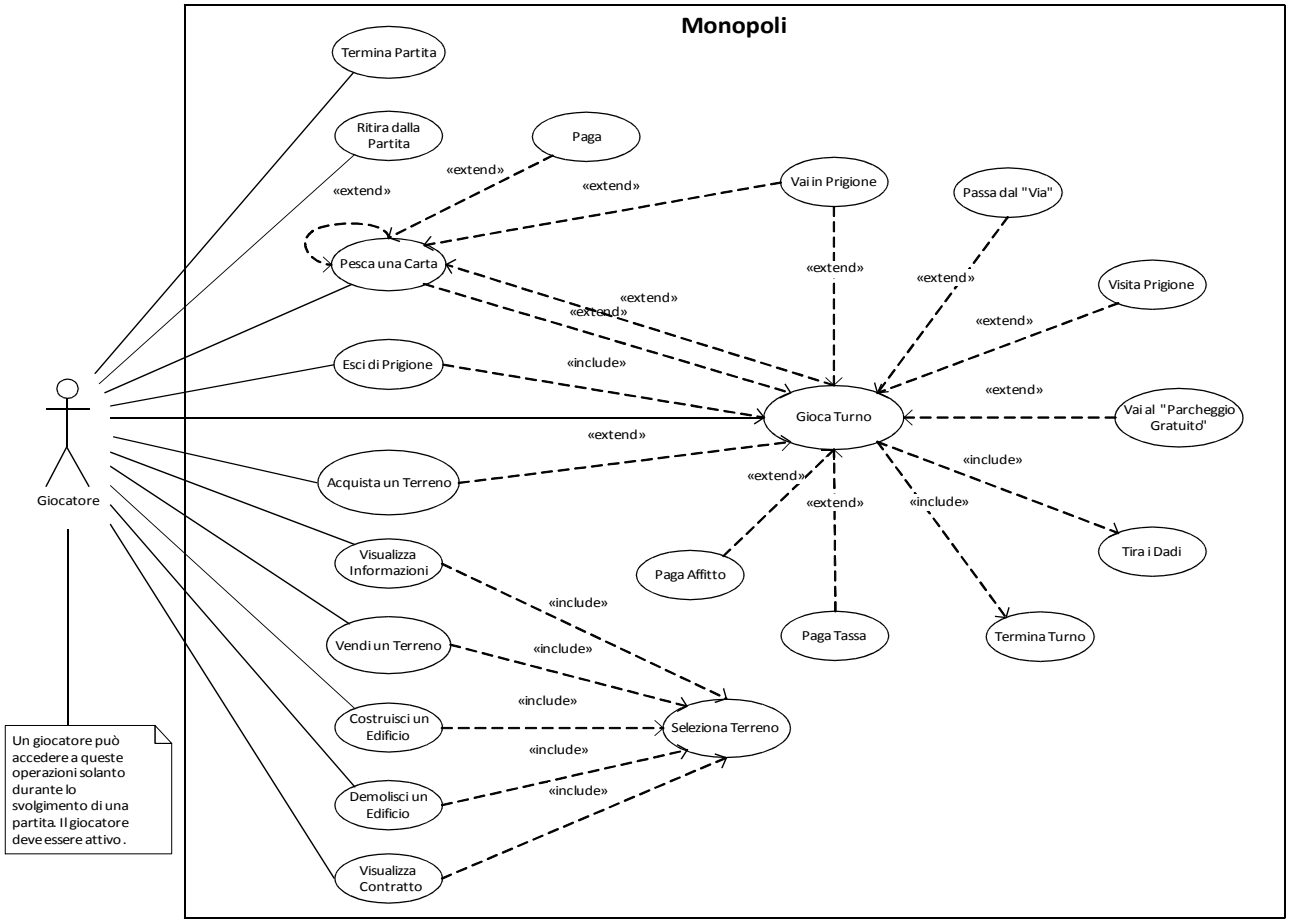
CASI D'USO

CREAZIONE PARTITA



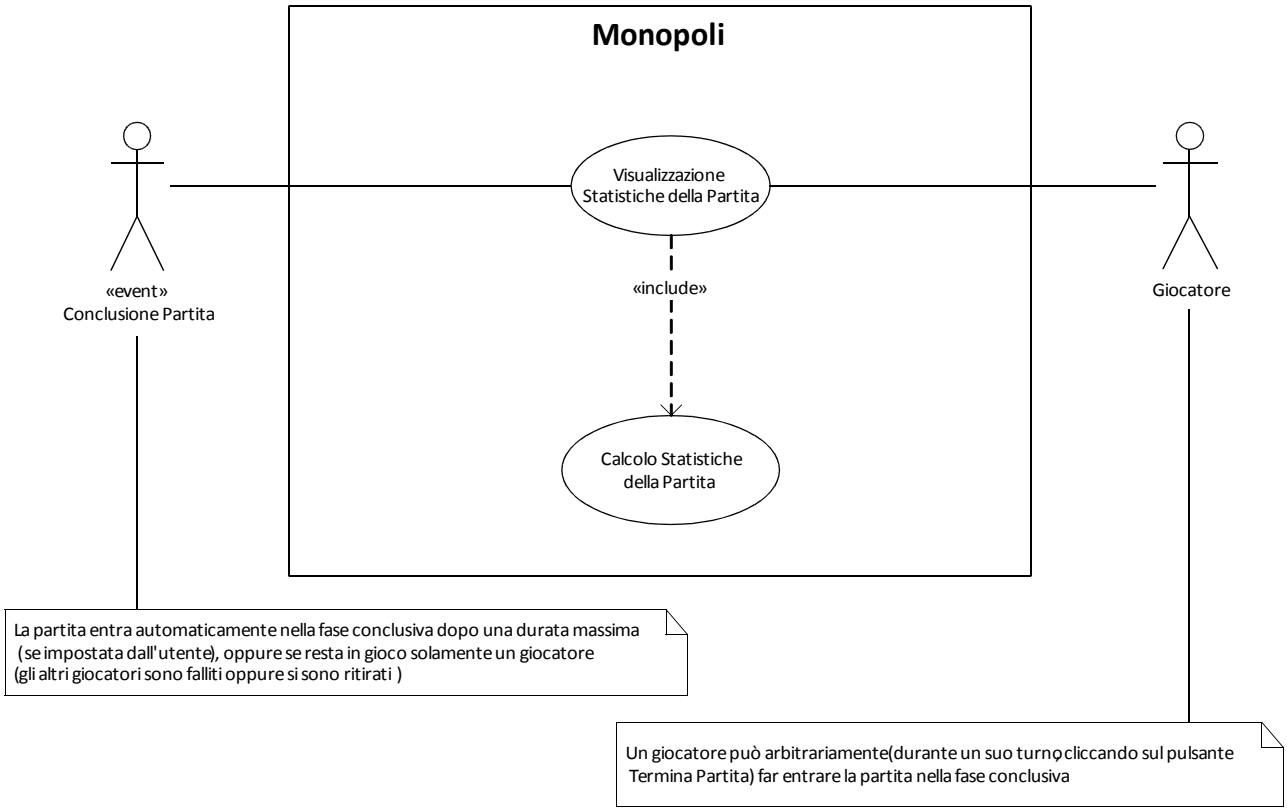
L'utente, una volta avviato il sistema, può scegliere di iniziare una Nuova Partita. Questo comporta una fase di Preparazione, in cui l'utente interagisce col sistema inserendo i dati dei giocatori ed un'eventuale durata massima per la partita, e una fase di Inizializzazione, gestita dal software. Nella seconda fase, vengono assegnati ad ogni giocatore dichiarato nella prima fase un capitale di partenza e dei contratti, in base al numero di giocatori partecipanti alla partita.
Infine, terminata anche la seconda fase, ad ogni giocatore vengono presentati i contratti ricevuti, il capitale di partenza e il capitale dopo aver pagato i caontratti.
Un Utente diventa Giocatore nel momento in cui inserisce i propri dati tra quelli dei partecipanti alla partita.
In qualsiasi momento durante la partita, gli utenti, giocatori e non, possono visualizzare il Regolamento del gioco.

SVOLGIMENTO PARTITA



Ogni Giocatore può, in qualsiasi momento durante lo svolgimento della partita, decidere di terminarla (vedi diagramma deguente).
Durante il proprio turno, invece, è obbligato ad effettuare determinate operazioni. Deve necessariamente tirare i dadi e, quando ha effettuato le operazioni obbligatorie ed eventiali operazioni facoltative, deve terminare esplicitamente il turno.
Le operazioni obbligatorie dipendono dallo stato del giocatore (ad esempio se è in prigione da 3 turni deve pagare la cauzione) e dalla sua posizione sulla tavola da gioco dopo aver spostato il proprio segnalino in seguito al lancio dei dadi se è in condizioni di farlo (ed esempio se si trova su una casella Tassa deve effettuare il pagamento della tassa specifica). Per tutte le operazioni di pagamento obbligatorie, è il Giocatore a stabilire quando eseguirle, tuttavia non può terminare il turno prima di aver saldato tutti i pagamenti. Analogamente avviene per l'operazione Pesca Carta, obbligatoria se il Giocatore si posa su una casella probabilità o imprevisti.
Infine, il Giocatore, sempre durante il proprio turno, può effettuare una serie di operazioni facoltative, tra le quali ritirarsi definitivamente dalla partita.
Un Giocatore inattivo non può più interagire con il sistema.

CONCLUSIONE PARTITA



In seguito alla decisione di terminare la partita da parte di uno dei partecipanti, allo scadere del tempo impostato come durata massima o quando vi è un solo Giocatore attivo, vengono visualizzate le statistiche della partita. Queste vengono prodotte in seguito ad un calcolo che determina una classifica dei Giocatori attivi, ritirati e falliti.

SCENARI

DESCRIZIONE DEI CASI D'USO

Note comuni:

- L'utente può in qualsiasi momento terminare una partita in corso, visualizzando così una schermata contenente le statistiche di fine partita.
- Se un giocatore non è in condizioni di compiere determinate operazioni, il sistema disabilita i pulsanti relativi ad esse. Esempio: se un giocatore capita su un terreno che non può acquistare (poiché già in possesso di un altro giocatore), il pulsante relativo all'acquisto di quel terreno non è selezionabile.
- Ogni qualvolta un giocatore si trova a dover effettuare il pagamento di una somma che eccede il suo attuale capitale, appare una notifica in cui si segnala al giocatore che effettuando tale pagamento fallirà e diventerà inattivo. Il giocatore può decidere se confermare l'operazione e fallire oppure annullarla ed effettuare altre operazioni prima di proseguire con quello specifico pagamento.
- Se un giocatore diventa inattivo, non potrà più in alcun modo interagire con il sistema.

CASI D'USO:

CU1 Nuova Partita
CU2 Preparazione
CU3 Inizializzazione Giocatori
CU4 Gioca Turno
CU5 Tira i Dadi
CU6 Passa dal “Via”
CU7 Vai in Prigione
CU8 Visita Prigione
CU9 “Parcheggio gratuito”
CU10 Paga Tassa
CU11 Paga Affitto
CU12 Acquista un Terreno
CU13 Vendi un Terreno
CU14 Costruisci un Edificio
CU15 Demolisci un Edificio
CU16 Visualizza Contratto
CU17 Seleziona Terreno
CU18 Esci di Prigione
CU19 Pesca una Carta
CU20 Paga
CU21 Visualizza Informazioni
CU22 Termina Turno
CU23 Termina Partita
CU24 Ritira dalla Partita
CU25 Visualizza Statistiche della Partita

Titolo	**CU1 – Nuova Partita**
Descrizione	Un Utente decide di cominciare una nuova Partita al gioco Monopoli
Relazioni	Preparazione ([CU2]), Inizializzazione Giocatori ([CU3])
Attori	Utente
Precondizioni	Non deve essere in corso un'altra Partita
Postcondizioni	La Partita è stata creata secondo i parametri inseriti dall'Utente
Scenario principale	1. A sistema avviato, l'Utente può decidere di iniziare una Nuova Partita, cliccando sull'apposito pulsante 2. Si apre una schermata di Preparazione in cui l'Utente deve inserire i dati necessari per creare la Partita ([CU2]) 3. Una volta confermati i dati inseriti nella fase di Preparazione, La Partita viene creata dal sistema ([CU3]) 4. Appare la Tavola da gioco della Partita appena creata, e i Giocatori possono da questo momento giocare a Monopoli
Scenari alternativi	3a. Se l'Utente decide di annullare la creazione della Partita, il sistema non creerà una Nuova Partita 3a.1 La schermata viene chiusa e l'Utente deve ripetere tutti i passaggi dal principio

Titolo	**CU2 – Preparazione**
Descrizione	L'Utente inserisce i dati necessari alla creazione di una Nuova Partita
Relazioni	Inserimento Durata Massima della Partita, Inserimento Giocatore
Attori	Utente
Precondizioni	- Una nuova Partita è in fase di creazione
Postcondizioni	Sono stati inseriti nome e Segnalino di almeno due Giocatori I nomi dei due Giocatori devono essere differenti
Scenario principale	1. Entrato nella fase di Preparazione, all'Utente appare un menu contenente un campo per l'immissione di una durata massima della Partita, e sei campi per l'inserimento dei Giocatori 2. L'Utente inserisce una durata massima 3. L'Utente inserisce i nomi dei Giocatori che parteciperanno alla Partita e seleziona per ognuno di essi uno specifico Segnalino 4. L'Utente conferma i propri inserimenti cliccando su un apposito pulsante
Scenari alternativi	2a. Se l'Utente non specifica una durata massima la partita avrà durata illimitata 4a. Se l'Utente non ha inserito le informazioni di almeno due Giocatori, il sistema non abilita il pulsante di conferma 4a.1 L'Utente torna al passo 3 4b. Se l' Utente inserisce in due campi differenti lo stesso nome, appare una notifica di errore e il sistema disabilita il pulsante di conferma 4b.1 L'Utente torna al passo 3

Titolo	**CU3 – Inizializzazione Giocatori**
Descrizione	Al termine della Preparazione della Partita, il sistema assegna automaticamente ai Giocatori un Capitale di partenza e dei Contratti
Relazioni	Assegnamento Capitali di Partenza, Distribuzione dei Contratti, Visualizza Contratti assegnati
Attori	Utente, Giocatore
Precondizioni	Si è conclusa con successo la fase di Preparazione della partita
Postcondizioni	Ogni Giocatore possiede dei Contratti ed un Capitale diverso da zero
Scenario principale	1. Il sistema assegna ad ogni Giocatore un capitale di partenza che dipende dal numero di Giocatori partecipanti alla Partita 2. Il sistema assegna ad ogni Giocatore dei Contratti, il cui numero dipende dal numero di partecipanti alla Partita, scelti in maniera casuale tra tutti quelli presenti nel gioco. Ogni Giocatore diventa così proprietario di alcuni Contratti 3. Per ogni Giocatore appare, a rotazione, una schermata in cui sono visualizzati i Contratti assegnati dal sistema, e il loro importo totale che il Giocatore dovrà pagare (l'importo verrà detratto dal Capitale di partenza), assieme a un pulsante di conferma 4. Ogni Giocatore clicca sul pulsante di conferma, pagando così automaticamente l'importo appena indicato 5. Quando tutti i Giocatori hanno confermato il pagamento dei Contratti, inizia la fase di svolgimento della Partita, in cui i Giocatori giocano dei Turni, a rotazione
Scenari alternativi	

Titolo	**CU4 – Gioca Turno**
Descrizione	Durante lo svolgimento del gioco, i Giocatori devono a rotazione giocare dei Turni. In ogni Turno devono tirare i dadi e, se sono nelle condizioni necessarie, ovvero non sono in Prigione, il sistema muove il loro Segnalino in base all'esito del tiro dei dadi. I Giocatori devono poi, prima di terminare il Turno, effettuare le operazioni previste dalla Casella di arrivo
Relazioni	Tira i Dadi ([CU5]), Passa dal Via ([CU6]), Vai in Prigione ([CU7]), Visita Prigione ([CU8]), “Parcheggio Gratuito” ([CU9]), Paga Tassa ([CU10]), Paga Affitto ([CU11]), Acquista un Terreno([CU12]), Pesca una Carta([CU19]), Termina Turno ([CU22])
Attori	Giocatore
Precondizioni	- È il Turno del Giocatore
Postcondizioni	Il Giocatore ha tirato i dadi, effettuato le operazioni necessarie e terminato il Turno
Scenario principale	Durante il suo Turno: 1. Il Giocatore Tira i dadi ([CU5]) 2. Il Giocatore attende che il sistema sposti il suo Segnalino in senso orario di un numero di Caselle pari al risultato ottenuto dal tiro dei dadi 3. La nuova posizione del Segnalino viene memorizzata dal sistema 4. La Casella sulla quale il Segnalino del Giocatore è stato mosso determina cosa succede al Giocatore durante quel Turno: 4.1 La Casella “Via” (o anche il semplice passaggio da questa Casella) corrisponde a [CU6] 4.2 La Casella “In prigione” corrisponde a [CU7], la Casella “Prigione” corrisponde a [CU8] 4.3 La Casella “Parcheggio gratuito” corrisponde a [CU9] 4.4 Le Caselle delle Tasse corrispondono a [CU10] 4.5 le Caselle dei Terreni possono corrispondere a [CU11] o [CU12], a seconda che siano possedute o meno da un altro Giocatore 4.6 le Caselle Probabilità e Imprevisti corrispondono a [CU19] 5. Il Giocatore può decidere di effettuare altre operazioni sempre possibili durante il Turno 6. Il Giocatore termina il proprio Turno ([CU22]) 7. Inizia il Turno del Giocatore successivo
Scenari alternativi	1a. Se il Giocatore si trova in Prigione può decidere di pagare la Cauzione prima di tirare i dadi 1b. Se il Giocatore viene dal caso d'uso Pesca Carta non deve tirare i dadi ma svolgere solamente le operazioni previste dalla Casella in cui è stato spostato a causa della Carta pescata; passa al punto 3 1c. Se il Giocatore viene dal caso d'uso Esci di Prigione e ha precedentemente tirato i dadi mantiene il tiro già effettuato 2a. Se il Giocatore si trova in Prigione da meno di tre Turni e non ha ottenuto un tiro di dadi doppio o non ha pagato la Cauzione non può muovere il proprio Segnalino e salta al passo 5 2b. Se il Giocatore si trova in Prigione da tre Turni e non ha ottenuto un tiro di dadi doppio, è costretto a uscire di Prigione pagaando la

	Cauzione 2b.1. Passa a [CU18] 2b.2. Il Giocatore procede comunque al passo 2, mantenendo il tiro di dadi già effettuato 2c. Se il Giocatore si trova in Prigione, ed ha ottenuto un tiro di dadi doppio, esce di Prigione e procede al passo 2 2d. Se il Giocatore non si trova in Prigione e ha ottenuto un tiro di dadi doppio per la terza volta consecutiva, finisce in Prigione 2d.1. Il Segnalino del Giocatore viene automaticamente spostato dal sistema sulla Casella Prigione, e il Giocatore si trova da quel momento in Prigione 2d.2. Il Giocatore salta al passo 5 6a. Se il Giocatore non si trova in Prigione e ha ottenuto un tiro di dadi doppio, deve rilanciare i dadi, tornando al passo 1

Titolo	**CU5 – Tira i Dadi**
Descrizione	Il Giocatore lancia due dadi, il cui risultato determinerà il movimento del Segnalino del Giocatore.
Relazioni	
Attori	Giocatore
Precondizioni	- È il Turno del Giocatore
Postcondizioni	
Scenario principale	1. Il Giocatore clicca sul pulsante Tira i Dadi per tirare i dadi 2. Appare il risultato dei due dadi appena lanciati. I valori dei due dadi lanciati appaiono singolarmente (ciascun dado ha sei facce)
Scenari alternativi	

Titolo	**CU6 – Passa dal “Via”**
Descrizione	Il Giocatore passa o si ferma sulla casella “Via” della Tavola da gioco
Relazioni	
Attori	Giocatore
Precondizioni	- È il Turno del Giocatore - Il Giocatore ha già lanciato i dadi - Il Giocatore si è fermato o è passato dalla Casella “Via”
Postcondizioni	
Scenario principale	1. Se il Giocatore passa, o finisce con il proprio Segnalino sulla Casella “Via”, riceve automaticamente un'indennità dalla Banca
Scenari alternativi	1a. Un Giocatore che finisce in Prigione non passa da questa Casella, e perciò non riceve nessuna indennità dalla Banca

Titolo	**CU7 – Vai in Prigione**
Descrizione	Il Giocatore finisce in Prigione, dalla quale potrà uscire solo in seguito a un tiro di dadi doppio oppure dopo aver pagato una Cauzione
Relazioni	
Attori	Giocatore
Precondizioni	- È il Turno del Giocatore - Il Giocatore ha già lanciato i dadi - Il Giocatore si è fermato sulla Casella “In Prigione”
Postcondizioni	Il Giocatore è in Prigione
Scenario principale	1. Appare un'indicazione che informa il Giocatore che il suo Segnalino verrà spostato in Prigione 2. Il Segnalino del Giocatore viene automaticamente spostato sulla Casella Prigione 3. Il Giocatore si trova da ora in Prigione
Scenari alternativi	

Titolo	**CU8 – Visita Prigione**
Descrizione	Il Giocatore si ferma in visita alla Prigione
Relazioni	
Attori	Giocatore
Precondizioni	- È il Turno del Giocatore - Il Giocatore ha già lanciato i dadi - Il Giocatore si è fermato sulla Casella “Prigione”
Postcondizioni	
Scenario principale	1. Il Giocatore visita la Prigione. Ai visitatori non accade nulla, pertanto il Giocatore non è da considerarsi in Prigione
Scenari alternativi	

Titolo	**CU9 – “Parcheggio gratuito”**
Descrizione	Il Giocatore finisce sul Parcheggio gratuito
Relazioni	
Attori	Giocatore
Precondizioni	- È il Turno del Giocatore - Il Giocatore ha già lanciato i dadi - Il Giocatore si è fermato sulla Casella “Parcheggio gratuito”
Postcondizioni	
Scenario principale	Il Giocatore si trova su un Parcheggio gratuito. Non gli accade nulla
Scenari alternativi	

Titolo	**CU10 – Paga Tassa**
Descrizione	Il Giocatore è costretto a pagare una Tassa alla Banca
Relazioni	
Attori	Giocatore
Precondizioni	- È il Turno del Giocatore - Il Giocatore ha già lanciato i dadi - Il Giocatore si è fermato sulla Casella “Tassa patrimoniale” oppure sulla Casella “Tassa di lusso”
Postcondizioni	
Scenario principale	1. Il Giocatore deve pagare una Tassa alla Banca, il cui importo è indicato sulla Casella stessa 2. Il Giocatore effettua il pagamento, cliccando sull'apposito pulsante
Scenari alternativi	2a. Se il Giocatore non ha un Capitale sufficiente per pagare la Tassa, fallisce. Il Giocatore può tuttavia vendere Terreni o Edifici alla Banca per incrementare il proprio Capitale ed essere così in grado di pagare la Tassa 2a.1 Se il Giocatore fallisce, tutte le sue proprietà passano automaticamente alla Banca, e il Giocatore diventa inattivo

Titolo	**CU11 – Paga Affitto**
Descrizione	Il Giocatore è costretto a pagare l'Affitto per sostare sulla proprietà di un altro Giocatore
Relazioni	
Attori	Giocatore
Precondizioni	- È il Turno del Giocatore - Il Giocatore ha già lanciato i dadi - Il Giocatore si è fermato su una Casella di tipo Terreno in possesso di un altro Giocatore
Postcondizioni	
Scenario principale	1. Il Giocatore deve pagare l'Affitto all'altro Giocatore, il cui importo dipende dallo specifico Terreno, dal numero di Terreni dello stesso Gruppo posseduti dal Giocatore avversario e dal numero di Edifici costruiti su quel Terreno. 2. Il Giocatore effettua il pagamento, cliccando sull'apposito pulsante
Scenari alternativi	2a. Se il Giocatore non ha un Capitale sufficiente per pagare l'Affitto, fallisce. Il Giocatore può tuttavia vendere Terreni o Edifici alla Banca per incrementare il proprio Capitale ed essere così in grado di pagare l'Affitto. 2a.1 Se il Giocatore fallisce, tutte le sue proprietà passano automaticamente alla Banca, e il Giocatore diventa inattivo 2a.2 La Banca paga automaticamente l'Affitto al Giocatore avversario in vece del Giocatore appena fallito

Titolo	**CU12 – Acquista un Terreno**
Descrizione	Se un Giocatore finisce su un Terreno posseduto dalla Banca, può decidere di acquistarlo, diventandone il proprietario
Relazioni	
Attori	Giocatore
Precondizioni	- È il Turno del Giocatore - Il Giocatore ha già lanciato i dadi - Il Giocatore si è fermato su una casella di tipo Terreno non posseduta da alcun Giocatore, quindi dalla Banca
Postcondizioni	Il sistema tiene traccia e registra l'acquisto (se avvenuto)
Scenario principale	Esistono tre tipi di Terreni acquistabili: i Terreni normali, le Stazioni ferroviarie e le Imprese di pubblica utilità 1. Il Giocatore può, in qualsiasi momento (prima della fine del Turno), decidere di acquistare il Terreno cliccando sul pulsante Acquista 2. Viene visualizzato l'importo che il Giocatore deve pagare per poter acquistare il Terreno, assieme a un pulsante di conferma dell'acquisto 3. Confermando, il Giocatore paga alla Banca l'importo relativo al Terreno da acquistare 4. Il Giocatore diventa possessore del Terreno e il sistema aggiorna lo stato del Giocatore (attivando eventualmente la possibilità di costruire Edifici su di esso)
Scenari alternativi	3a. Se il Giocatore non ha un Capitale abbastanza alto da poter acquistare il Terreno, gli viene notificato che procedendo fallirà 3a.1 Se il Giocatore procede comunque, egli fallisce 3a.2 Se il Giocatore rinuncia all'acquisto non capita nulla

Titolo	**CU13 – Vendi un Terreno**
Descrizione	Un Giocatore ha la possibilità di vendere alla Banca dei Terreni da lui posseduti, per metà dell'importo pagato al momento dell'acquisto del Terreno stesso
Relazioni	Seleziona Terreno ([CU17])
Attori	Giocatore
Precondizioni	- È il Turno del Giocatore - Il Giocatore possiede almeno un Terreno
Postcondizioni	Il sistema registra l'avvenuta vendita, e il Giocatore non è più possessore di quello specifico Terreno, che è ora in possesso della Banca Il Capitale del Giocatore aumenta dell'importo ricevuto dalla Banca al momento della vendita
Scenario principale	1. Il Giocatore può, in qualsiasi momento (prima della fine del Turno), decidere di vendere un Terreno in suo possesso cliccando sul pulsante Vendi Terreno 2. Al Giocatore viene richiesto di selezionare un Terreno tra quelli in suo possesso (CU17) 3. Viene visualizzato l'importo che il Giocatore riceverà dalla Banca vendendole quel Terreno, assieme a un pulsante di conferma 4. Confermando, il Giocatore riceve automaticamente l'importo relativo al Terreno venduto alla Banca 5. Il Giocatore non è più possessore di quel Terreno e il sistema aggiorna lo stato del Giocatore (revocandogli la possibilità di visualizzare il Contratto di quello specifico Terreno e di costruire Edifici su di esso)
Scenari alternativi	

Titolo	**CU14 – Costruisci un Edificio**
Descrizione	Un Giocatore ha la possibilità di costruire sui propri Terreni degli Edifici, pagandone il relativo prezzo alla Banca. La presenza di un Edificio su un Terreno ne incrementa l'affitto che altri Giocatori devono pagare in caso di fermata su quel Terreno
Relazioni	Seleziona Terreno ([CU17])
Attori	Giocatore
Precondizioni	- È il Turno del Giocatore - Il Giocatore possiede tutti i Terreni normali di uno stesso Gruppo
Postcondizioni	Il sistema registra l'avvenuta costruzione
Scenario principale	1. Il Giocatore può, in qualsiasi momento (prima della fine del Turno), decidere di costruire un Edificio su un Terreno in suo possesso cliccando sul pulsante Costruisci un Edificio 2. Al Giocatore viene richiesto di selezionare un Terreno tra quelli al momento edificabili in suo possesso ([CU17]) 3. Viene visualizzato l'importo totale che il Giocatore deve pagare alla Banca per costruire l'Edificio, assieme a un pulsante di conferma 4. Confermando, il Giocatore paga automaticamente il prezzo di costruzione alla Banca 5. Gli Edifici appena costruiti appaiono sul Terreno del Giocatore 5.1 Se sul Terreno erano presenti meno di 4 Edifici viene eretta una casa 5.2 Se sul Terreno erano presenti 4 case viene eretto un albergo
Scenari alternativi	4a. Se il Giocatore non ha un Capitale sufficiente per poter acquistare l'Edificio, gli viene notificato che procedendo fallirà 4a.1 Se il Giocatore procede comunque, egli fallisce 4a.2 Se il Giocatore rinuncia all'acquisto non capita nulla

Titolo	**CU15 – Demolisci un Edificio**
Descrizione	Un Giocatore ha la possibilità di vendere alla Banca un Edificio precedentemente costruito su una proprietà, per metà del prezzo pagato al momento della costruzione dell'Edificio stesso
Relazioni	Seleziona Terreno ([CU17])
Attori	Giocatore
Precondizioni	- È il Turno del Giocatore - Il Giocatore possiede almeno un Edificio
Postcondizioni	Il sistema registra l'avvenuta demolizione Il Capitale del Giocatore aumenta dell'importo ricevuto dalla Banca al momento della vendita
Scenario principale	1. Il Giocatore può, in qualsiasi momento (prima della fine del Turno), decidere di demolire un Edificio su un Terreno in suo possesso cliccando sul pulsante Demolisci un Edificio 2. Al Giocatore viene richiesto di selezionare un Terreno tra quelli in suo possesso su cui al momento è costruito almeno un Edificio ([CU17]) 3. Viene visualizzato l'importo totale che la Banca pagherà al Giocatore per la demolizione dell'Edificio, assieme a un pulsante di conferma 4. Confermando, il Giocatore riceve automaticamente l'importo relativo all'Edificio demolito dalla Banca 5. L'Edificio appena demolito scompare dal Terreno del Giocatore
Scenari alternativi	

Titolo	**CU16 – Visualizza Contratto**
Descrizione	Il Giocatore può vedere le informazioni contenute in un Contratto, relative a un Terreno in suo possesso
Relazioni	Seleziona Terreno ([CU17])
Attori	Giocatore
Precondizioni	- È il Turno del Giocatore - Il Giocatore possiede almeno un Terreno
Postcondizioni	
Scenario principale	1. Il Giocatore può, in qualsiasi momento (prima della fine del Turno), decidere di visualizzare il Contratto di un Terreno in suo possesso cliccando sul pulsante Visualizza contratto 2. Al Giocatore viene richiesto di selezionare un Terreno tra quelli in suo possesso ([CU17]) 3. Appare il Contratto relativo al Terreno selezionato, con tutte le relative informazioni
Scenari alternativi	

Titolo	**CU17 – Seleziona Terreno**
Descrizione	Il Giocatore seleziona un Terreno
Relazioni	
Attori	Giocatore
Precondizioni	- È il Turno del Giocatore
Postcondizioni	
Scenario principale	1. Si apre un menu contenente i nomi dei Terreni 2. Il Giocatore seleziona uno dei Terreni elencati e conferma la propria selezione
Scenari alternativi	1a. Se il Giocatore sta costruendo un Edificio, vengono visualizzati solamente i Terreni posseduti dal Giocatore su cui è possibile costruire un Edificio 1b. Se il Giocatore sta demolendo un Edificio, vengono visualizzati solamente i Terreni posseduti dal Giocatore sui quali è correntemente costruito un Edificio 1c. Se il Giocatore sta vendendo un Terreno, vengono visualizzati i Terreni in possesso del Giocatore, ma non vengono visualizzati i Terreni di un Gruppo, sulle cui Caselle è stato costruito in totale almeno un Edificio 1d. Se il Giocatore vuole visualizzare un Contratto, vengono visualizzati solamente i Terreni in possesso del Giocatore

Titolo	**CU18 – Esci di Prigione**
Descrizione	Un Giocatore che si trova in Prigione paga una Cauzione per uscire
Relazioni	Gioca Turno([CU4])
Attori	Giocatore
Precondizioni	- È il Turno del Giocatore - Il Giocatore è in Prigione
Postcondizioni	Il Giocatore non è più in Prigione
Scenario principale	1. Il Giocatore può decidere di uscire di Prigione tramite l'apposito pulsante. Così facendo paga automaticamente una Cauzione alla Banca. 2. Dopo aver pagato la Cauzione il Giocatore non è più in Prigione e procede giocando normalmente il suo Turno, al [CU4]
Scenari alternativi	1a. Se il Giocatore si trova in Prigione da tre Turni e non ha ottenuto un tiro di dadi doppio, è costretto a pagare la Cauzione alla Banca e a uscire di Prigione.

Titolo	**CU19 – Pesca una Carta**
Descrizione	Il Giocatore finisce su una casella Probabilità o Imprevisti e pesca dunque una carta, contenente delle istruzioni, che il Giocatore dovrà seguire alla lettera
Relazioni	Paga([CU20]), Vai in Prigone ([CU7]), GiocaTurno([CU4])
Attori	Giocatore
Precondizioni	- È il Turno del Giocatore - Il Giocatore ha tirato i dadi - Il Giocatore si è fermato su una Casella di tipo “Probabilità” o “Imprevisti”
Postcondizioni	Il sistema registra le operazioni effettuate dal Giocatore
Scenario principale	Esistono due mazzi di carte: le Probabilità e gli Imprevisti. La Casella sulla quale si è fermato e le istruzioni della carta discriminano le operazione che il Giocatore deve effettuare 1. Il Giocatore pesca una carta cliccando sul pulsante Pesca carta 2. Appare la relativa carta pescata, contenente le istruzioni da seguire 3. Il Giocatore esegue le istruzioni cliccando sull'apposito pulsante che viene abilitato una volta pescata la carta. Le possibili istruzioni possono prevedere: 3.1 Un pagamento di un determinato importo [CU20] 3.2 Uno spostamento su un'altra Casella [CU4] 3.3 La ricezione di un importo di denaro 3.4 La condanna alla Prigione [CU7] 3.5 La pesca di un'altra Carta ripartendo dal punto 1 4. Lo stato del Giocatore viene aggiornato 5. Una volta seguite le istruzioni, la carta viene rimessa automaticamente in fondo al relativo mazzo
Scenari alternativi	3a. Se la Carta pescata prevede un versamento di denaro al Giocatore o un suo spostamento sulla Tavola daGioco, l'operazione avviene in automatico

Titolo	**CU20 – Paga**
Descrizione	Il Giocatore paga una somma di denaro imposta dalle istruzioni di una carta pescata.
Relazioni	
Attori	Giocatore
Precondizioni	- È il Turno del Giocatore - Il Giocatore ha tirato i dadi - Il Giocatore si è fermato su una Casella di tipo “Probabilità” o “Imprevisti” - Il Giocatore ha già pescato una carta che prevede un pagamento
Postcondizioni	
Scenario principale	1. Il Giocatore clicca sull'apposito pulsante per effettuare il pagamento previsto dalla carta.
Scenari alternativi	1.2a. Se il Giocatore non ha un Capitale sufficiente per effettuare un pagamento previsto dalla carta pescata, fallisce. Il Giocatore può tuttavia vendere Terreni o Edifici alla Banca per incrementare il proprio Capitale ed essere così in grado di effettuare il pagamento. 1.2a.1 Se il Giocatore fallisce, tutte le sue proprietà passano automaticamente alla banca, e il Giocatore diventa inattivo 1.2a.2 La Banca paga automaticamente, in vece del Giocatore fallito, eventuali altri Giocatori, che avrebbero dovuto ricevere un pagamento, in base alle istruzioni della carta pescata

Titolo	**CU21 – Visualizza Informazioni**
Descrizione	Il Giocatore vuole ottenere le informazioni relative a uno specifico Terreno
Relazioni	Seleziona Terreno ([CU17])
Attori	Giocatore
Precondizioni	- È il Turno del Giocatore
Postcondizioni	
Scenario principale	1. In qualunque momento durante il suo Turno, il Giocatore può decidere di visualizzare le informazioni di un Terreno cliccando sul pulsante Visualizza informazioni 2. Al Giocatore viene richiesto di selezionare un Terreno tra tutti quelli esistenti ([CU17]) 3. Appaiono le informazioni relative al Terreno selezionato, contenenti l'attuale possessore del Terreno e l'importo di acquisto del Terreno qualora esso sia in possesso della Banca
Scenari alternativi	

Titolo	**CU22 – Termina Turno**
Descrizione	Il Giocatore decide di terminare il proprio Turno e far giocare così un Turno a un altro Giocatore.
Relazioni	
Attori	Giocatore
Precondizioni	- È il Turno del Giocatore - Il Giocatore ha tirato i dadi - Il Giocatore ha eseguito tutte le operazioni obbligatorie specificate dalla Casella sulla quale si è ritrovato dopo aver tirato i dadi
Postcondizioni	Il sistema ha registrato le operazioni effettuate dal Giocatore e ne ha aggiornato lo stato Il Giocatore non può più interagire con il sistema fino al suo prossimo Turno
Scenario principale	1. Il Giocatore, una volta effettuate tutte le operazioni obbligatorie del Turno, clicca sul pulsante Termina Turno 2. Il Turno del Giocatore termina, e da quel momento non può più effettuare alcun tipo di operazione fino al suo prossimo Turno
Scenari alternativi	

Titolo	**CU23 – Termina Partita**
Descrizione	Terminazione arbitraria e irreversibile della Partita in corso
Relazioni	Visualizza Statistiche della Partita (CU25)
Attori	Giocatore
Precondizioni	- È il Turno del Giocatore
Postcondizioni	La Partita è conclusa, pertanto nessun Giocatore potrà più giocare dei Turni
Scenario principale	1. In qualunque momento durante il suo Turno, il Giocatore può decidere di terminare la Partita cliccando sul pulsante Termina Partita 2. La Partita viene conclusa e il sistema calcolerà le statistiche della Partita stessa, dichiarando un Vincitore sulla base di esse
Scenari alternativi	

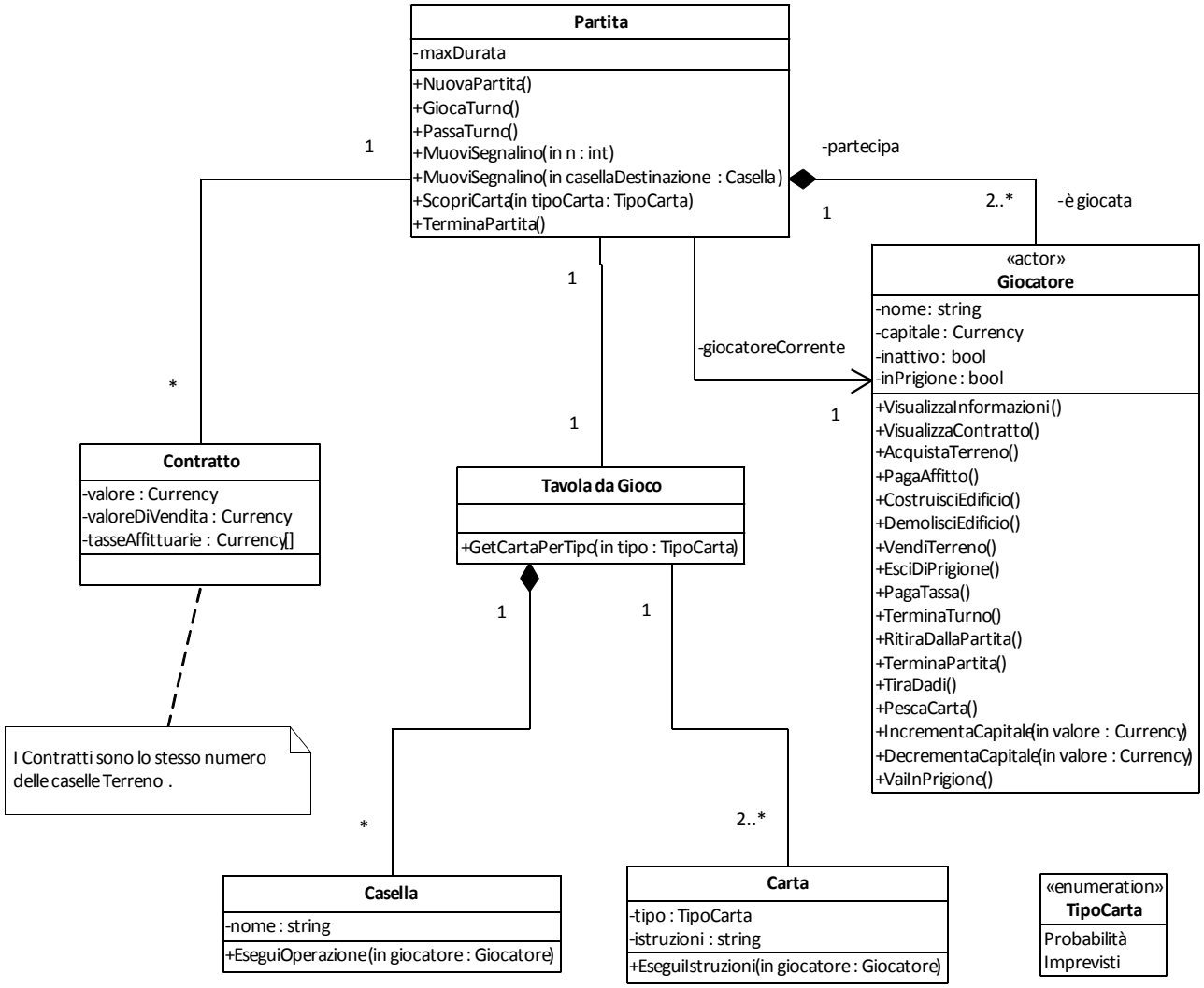
Titolo	**CU24 – Ritira dalla Partita**
Descrizione	Un Giocatore si ritira dalla Partita corrente
Relazioni	Visualizza Statistiche della Partita (CU25)
Attori	Giocatore
Precondizioni	- È il Turno del Giocatore
Postcondizioni	Il Giocatore è inattivo e non potrà più giocare dei Turni
Scenario principale	In qualunque momento durante il suo Turno, il Giocatore può decidere di ritirarsi dalla Partita. 1. Il Giocatore clicca sul pulsante Ritira dalla Partita, rinunciando a giocare il Turno corrente e tutti i suoi Turni successivi. Tutte le sue proprietà passano automaticamente alla Banca 2. Il Giocatore diventa inattivo e non potrà più interagire con il sistema 3. Inizia il Turno del Giocatore successivo
Scenari alternativi	2a. Se il Giocatore si ritira prima di aver pagato l'affitto a un altro Giocatore, la Banca provvederà automaticamente a pagare l'affitto a quel Giocatore in vece del Giocatore appena ritiratosi 3a. Se il Giocatore successivo è l'ultimo rimasto attivo, la partita termina e il sistema calcola le statistiche

Titolo	**CU25 – Visualizza Statistiche della Partita**
Descrizione	Al termine della Partita, quando si entra nella fase di Conclusione, il sistema presenta all'Utente una schermata di statistiche relative alla Partita giocata e decreta il Vincitore della Partita
Relazioni	Calcolo Statistiche della Partita
Attori	Giocatore
Precondizioni	- La Partita è entrata in fase di Conclusione, in seguito a due possibili eventi: a) È scaduto il tempo della durata massima della Partita b) È rimasto in gioco soltanto un Giocatore, e tutti gli altri sono inattivi oppure: c) Un Giocatore ha deciso di terminare arbitrariamente la Partita
Postcondizioni	La Partita è conclusa e non è più in alcun modo accessibile
Scenario principale	1. Il sistema effettua un calcolo dei possedimenti totali dei Giocatori, sia attivi che inattivi, che comprende il loro Capitale, i Terreni e gli Edifici posseduti 2. Il sistema costruisce una classifica sulla base del calcolo appena effettuato, in cui vengono presentati i Giocatori in ordine decrescente in base ai loro possedimenti totali 3. All'Utente appare una finestra contente la classifica e altre informazioni calcolate dal sistema 4. Il Giocatore in cima alla classifica è il Vincitore della Partita 5. L'Utente clicca su un pulsante di conferma per chiudere definitivamente la Partita
Scenari alternativi	

ANALISI DEL DOMINIO – MODELLO DEI DATI

DIAGRAMMI DELLE CLASSI DI ANALISI

PARTITA



Partita: rappresenta l'insieme di tutto ciò che fa parte del gioco: carte, tavola da gioco, contratti, giocatori. Si occupa, infatti, di svolgere i servizi che prevedono l'interazione di più parti del gioco come lo spostamento dei segnalini sulla tavola. Inoltre mantiene traccia del giocatore che sta attualmente giocando il suo turno.
Attributi:

- maxDurata: durata massima impostata per la partita.

Operazioni:

- NuovaPartita(): assegna ad ogni giocatore il segnalino, il capitale e il numero di contratti che gli spettano ad inizio partita; stabilisce inoltre l'ordine in cui i giocatori giocano il proprio turno.
- GiocaTurno(): il giocatore corrente esegue le operazioni previste dal suo turno di gioco.
- PassaTurno(): il giocatore successivo diventa quello corrente nella lista dei giocatori partecipanti alla partita.
- MuoviSegnalino(in n : int): muove il segnalino associato al giocatore che sta giocando il proprio turno di n caselle sulla tavola da gioco, partendo dalla casella sulla quale si trova; il valore di n può essere, per esempio, il risultato del lancio dei dadi.
- MuoviSegnalino(in casellaDestinazione : Casella): muove il segnalino associato al giocatore che sta giocando il proprio turno sulla casella di destinazione specificata come argomento. Operazione effettuata in caso di esecuzione di un'operazione di una carta Imprevisti che prevede lo spostamento del giocatore su una determinata casella (es. In prigione).
- ScopriCarta(in tipo : TipoCarta): fornisce una Carta del tipo descritto, fra le carte di quel tipo presenti sulla tavola da gioco.
- TerminaPartita(): la partita termina e si passa alla visualizzazione delle statistiche.

Tavola da Gioco: è formata da caselle e contiene le carte. È univoca per una partita.
Operazioni:

- GetCartaPerTipo(in tipo : TipoCarta): ritorna la Carta in cima al mazzo del tipo corrispondente al tipo di casella che gli viene passato (es tipoCarta= Imprevisti, ritorna una carta imprevisti). Vedi diagramma di sequenza.

Casella: la tavola da gioco è composta da n caselle.
Attributi:

- nome: è il nome della specifica casella visualizzato sulla casella stessa nella tavola da gioco.

Operazioni:

- EseguiOperazione(in giocatore : Giocatore): la casella esegue un'operazione sul giocatore passato come argomento.
  ES. Se la Casella è un Terreno in possesso di un altro giocatore, questa fa pagare l'affitto al giocatore passato come argomento. Se non è prevista nessuna operazione, l'esecuzione del metodo non avrà effetti.

Carta: Nel gioco deve essere presente almeno una carta per tipo.
Attributi:

- tipo: le carte sono di tipo Probabilità o Imprevisti;
- istruzioni: descrizione di cosa fa la carta.

Operazioni:

- EseguiIstruzioni(in giocatore : Giocatore): la carta esegue le operazioni previste dalle istruzioni, sul giocatore passato come argomento.

Contratto: rappresenta i cartoncini contratto del gioco. In una partita devono essere presenti tanti contratti quante sono le proprietà (i terreni). I contratti contengono tutte le informazioni relative al terreno a cui corrispondono.
Ogni giocatore può possedere tanti contratti quanti riesce ad acquistarne. Ogni contratto può appartenere ad un giocatore oppure alla banca.
Attributi:

- valore: valore di acquisto del terreno associato al contratto;
- valoreDiVendita: somma che si ricava vendendo il terreno associato al contratto;
- tasseAffittuarie: somme che devono pagare i giocatori che finiscono su questa casella, nel caso in cui appartenga ad un altro giocatore; il valore varia a seconda del numero di edifici presenti sul terreno corrispondente e a seconda del numero di terreni del medesimo gruppo posseduti dal proprietario di questo contratto (associato a un terreno).

Carte e contratti, così come tavola da gioco, restano presenti come entità fisiche e vengono riutilizzati in partite successive.

Giocatore: è l'entità che rappresenta l'utente partecipante alle partita in corso. Contiene il nome inserito in fase di preparazione della partita, il capitale attuale del giocatore e il suo stato (inattivo, in prigione). Raccoglie tutte le operazioni che un giocatore può effettuare. Il Giocatore è tale solamente durante lo svogimento della partita; il suo ruolo decade nel momento in cui la partita viene terminata. Questo giustifica l'uso di un'associazione di tipo composizione con Partita.

Attributi:

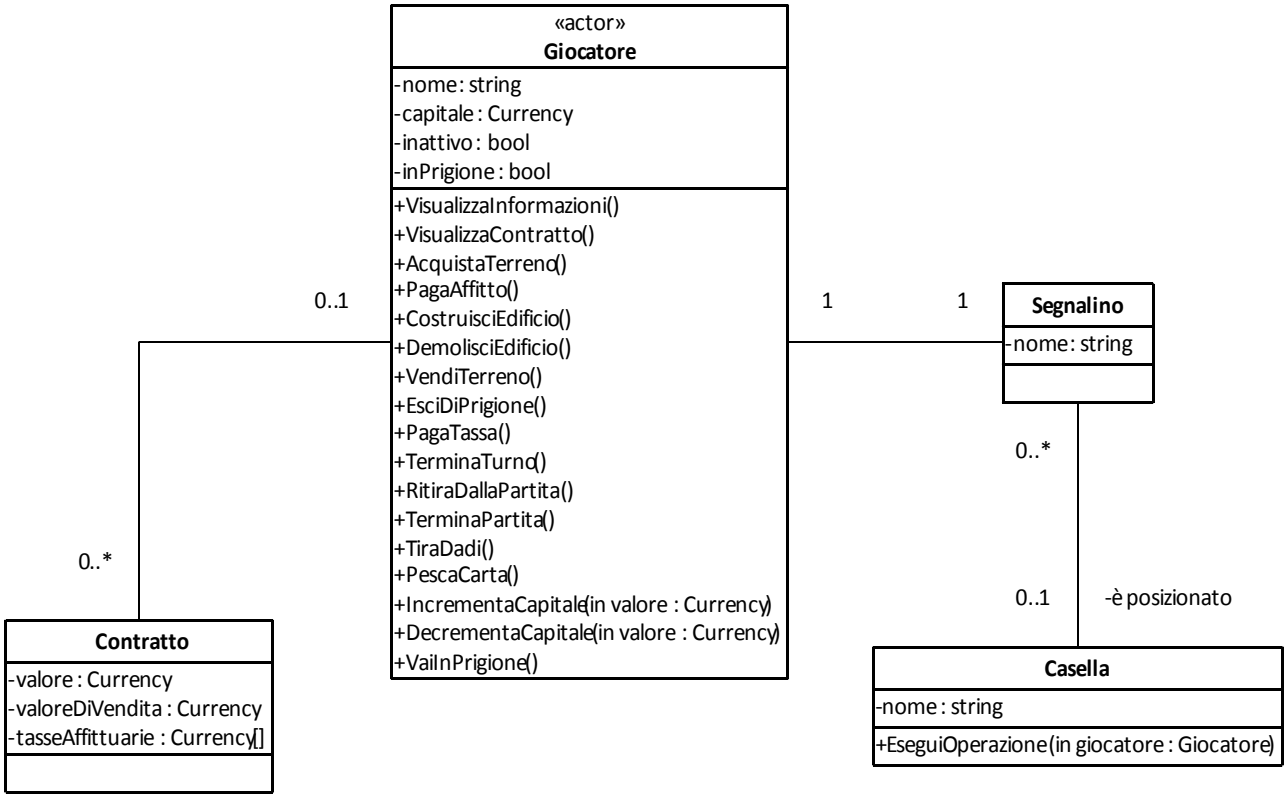
- nome: nome del giocatore;
- capitale: denaro a disposizione del giocatore;
- inattivo: attributo che descrive lo stato del giocatore; se egli è inattivo, non può più interagire con la partita in corso (non gioca più turni). Quando un giocatore perde (fallimento) o si ritira dal gioco, viene posto inattivo e non cancellato in modo tale che possa essere considerato nel calcolo delle statistiche alla fine della partita;
- InPrigione: attributo che indica se un giocatore si trova attualmente in prigione o meno; un giocatore in prigione deve comportarsi secondo le regole imposte da questo status (vedi casi d'uso).

Operazioni:

- VisualizzaInformazioni(): permette di visualizzare le informazioni relative a qualsiasi terreno;
- VisualizzaContratto(): si visualizzano i contratti posseduti dal giocatore;
- AcquistaTerreno(): se il terreno su cui il giocatore si trova è acquistabile, egli lo acquista;
- PagaAffitto(): il giocatore paga l'affitto al proprietario del terreno su cui si trova;
- CostruisciEdificio(): permette di costruire un edificio su uno dei terreni posseduti dal giocatore, a patto che ci siano le condizioni per farlo, ovvero che il giocatore possieda tutti i terreni del gruppo del terreno prescelto;
- DemolisciEdificio(): permette di demolire un edificio costruito in precedenza, ricavandone la somma corrispondente;
- VendiTerreno(): permette di vendere uno dei terreni posseduti;
- EsciDiPrigione(): il giocatore che è in prigione ne esce pagando la cauzione;
- PagaTassa(): il giocatore paga una tassa;
- TerminaTurno(): il giocatore conclude il suo turno, la Partita passerà il turno al giocatore successivo;
- RitiraDallaPartita(): il giocatore diventa inattivo, non giocherà più turni fino alla fine della partita;
- TiraDadi(): il giocatore simula il lancio di due dadi a sei facce. Il sistema mostra il numero di posizioni di cui il giocatore si deve muovere sulla tavola da gioco, muove il segnalino corrispondente sulla nuova casella e imposta la nuova posizione del giocatore sulla casella di arrivo (vedi diagramma di sequenza);
- PescaCarta() : ritorna una carta di tipo Imprevisti o Probabilità a seconda della posizione corrente del giocatore (vedi diagramma di sequenza);
- IncrementaCapitale(in valore: Currency): il capitale viene incrementato del valore passato come argomento;
- DecrementaCapitale(in valore: Currency): il capitale viene decrementato del valore passato come argomento;
- VaiInPrigione(): lo stato del giocatore è aggiornato in "in Prigione" (vedi diagramma degli stati).

GIOCATORE

I giocatori si muovono sulle caselle dalla tavola da gioco, rappresentati dal proprio segnalino.
Ogni giocatore può acquistare una serie di contratti.

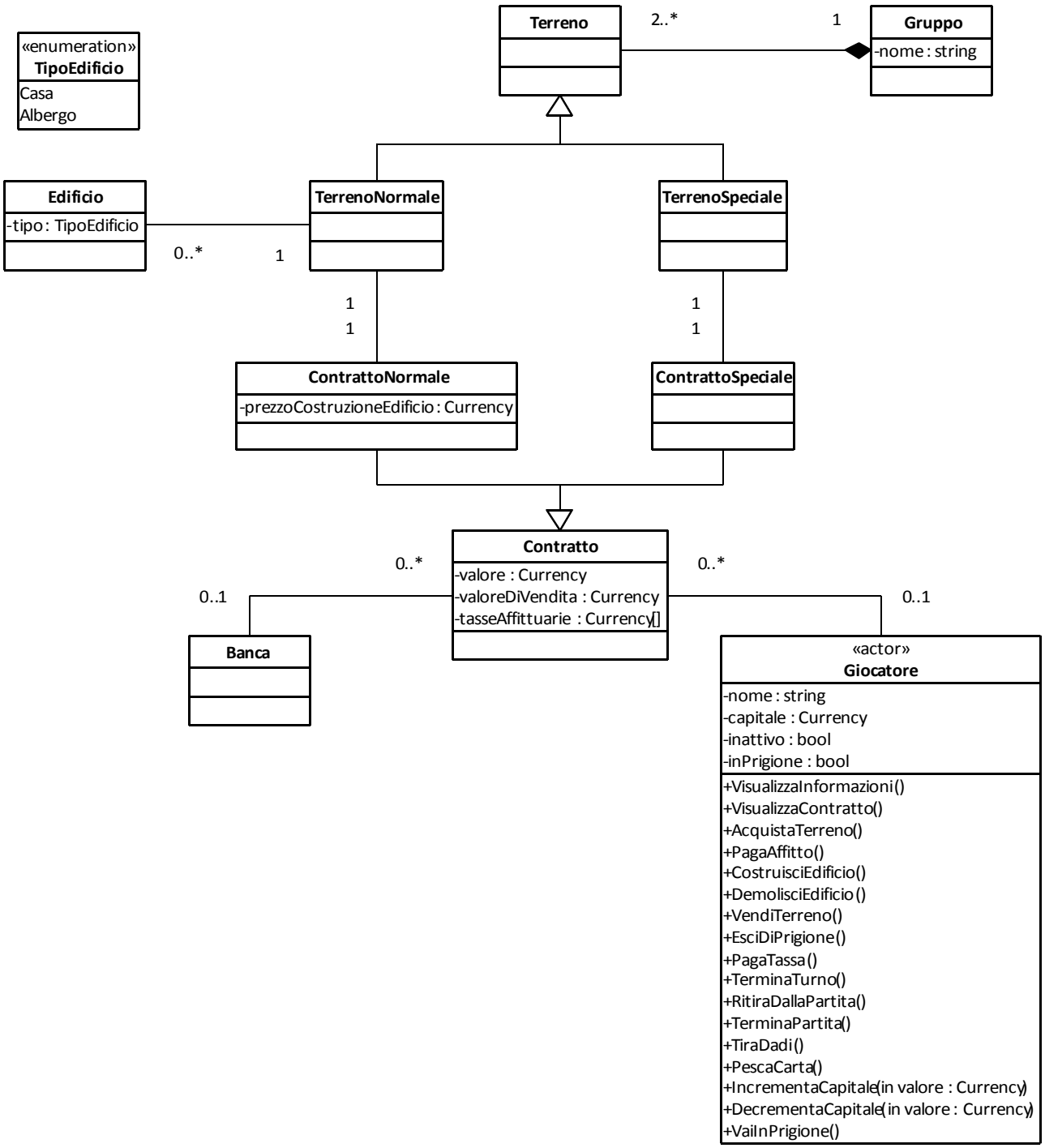


Segnalino: è l'entità che rappresenta univocamente un giocatore sulla tavola da gioco. Esso è posizionato, in ogni istante, su una sola casella a meno che il giocatore non sia inattivo. In questo caso il segnalino viene rimosso dalla tavola. Su una casella possono essere presenti più segnalini nello stesso momento senza alcuna implicazione.
Attributi:

- nome: è il nome del segnalino.

TERRENO e CONTRATTO

I terreni si suddividono in terreni normali, ovvero edificabili, e terreni speciali, cioè società di pubblica utilità e stazioni ferroviarie. Ogni terreno appartiene ad un gruppo.
Sui terreni normali possono essere costruite delle case o degli alberghi.
Ad ogni terreno è associato un contratto che ne fornisce tutte le informazioni.
I contratti possono essere acquistati dai giocatori; se un contratto non appartiene a nessun giocatore, allora è di proprietà della banca.



Terreno: rappresenta tutte le caselle di tipo proprietà, ovvero quelle che possono essere acquistate e che hanno associato un contratto. Un terreno può essere **normale**, ovvero edificabile, oppure **speciale**, come società e stazioni.

Contratto: rappresenta i cartoncini presenti nel gioco in scatola che contengono tutte le informazioni relative ai terreni, come prezzo, valore di vendita e affitto. Possono essere di due tipi, in base alla tipoloigia di proprietà cui sono associati. I contratti possono appartenere alla banca oppure ad un giocatore.

Contratto Normale:
Attributo:

- prezzoCostruzioneEdificio: prezzo di costruzione di un edificio sul terreno associato al contratto.

ContrattoSpeciale: non contiene i prezzi di costruzione degli edifici in quanto non è possibile costruire edifici su terreni speciali.

Edificio: rappresenta un Edificio che può essere eretto sui terreni edificabili.
Attributo:

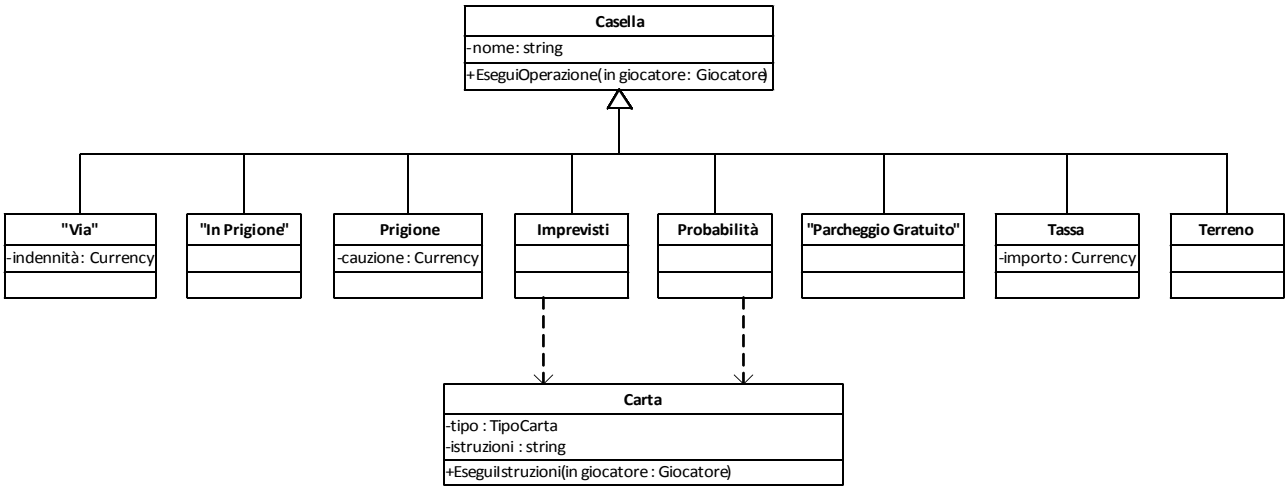
- tipo: rappresenta il tipo dell'edificio (Casa o Albergo).

Gruppo: è la categoria nella quale rientra ogni terreno.
Attributo:

- nome: nome del gruppo.

CASELLA

Vi sono diversi tipi di caselle in base ai quali viene definito il comportamento del giocatore che vi si posa sopra. L'arrivo sulle caselle Imprevisti e Probabilità obbliga il giocatore a pescare la rispettiva carta e a seguirne le istruzioni.



Via: è la casella di partenza per tutti i giocatori. Ogni volta che vi si transita o vi si ferma sopra un giocatore, questo riceve un'indennità, ovvero una somma di denaro indicata nell'attributo omonimo.
Attributo:

- indennità: somma che un giocatore riceve se sosta o transita dalla casella Via.

In Prigione: è una casella che obbliga il giocatore che vi capita sopra ad andare in prigione.

Prigione: è una casella che prevede un duplice comportamento del giocatore. Se esso vi finisce sopra in seguito ad un lancio dei dadi, non subisce nessuna conseguenza; se, invece, viene mandato in prigione secondo una delle modalità descritte nel regolamento, allora deve rimanere su questa casella finchè si trova imprigionato.
Attributo:

- cauzione: somma da pagare per uscire di prigione.

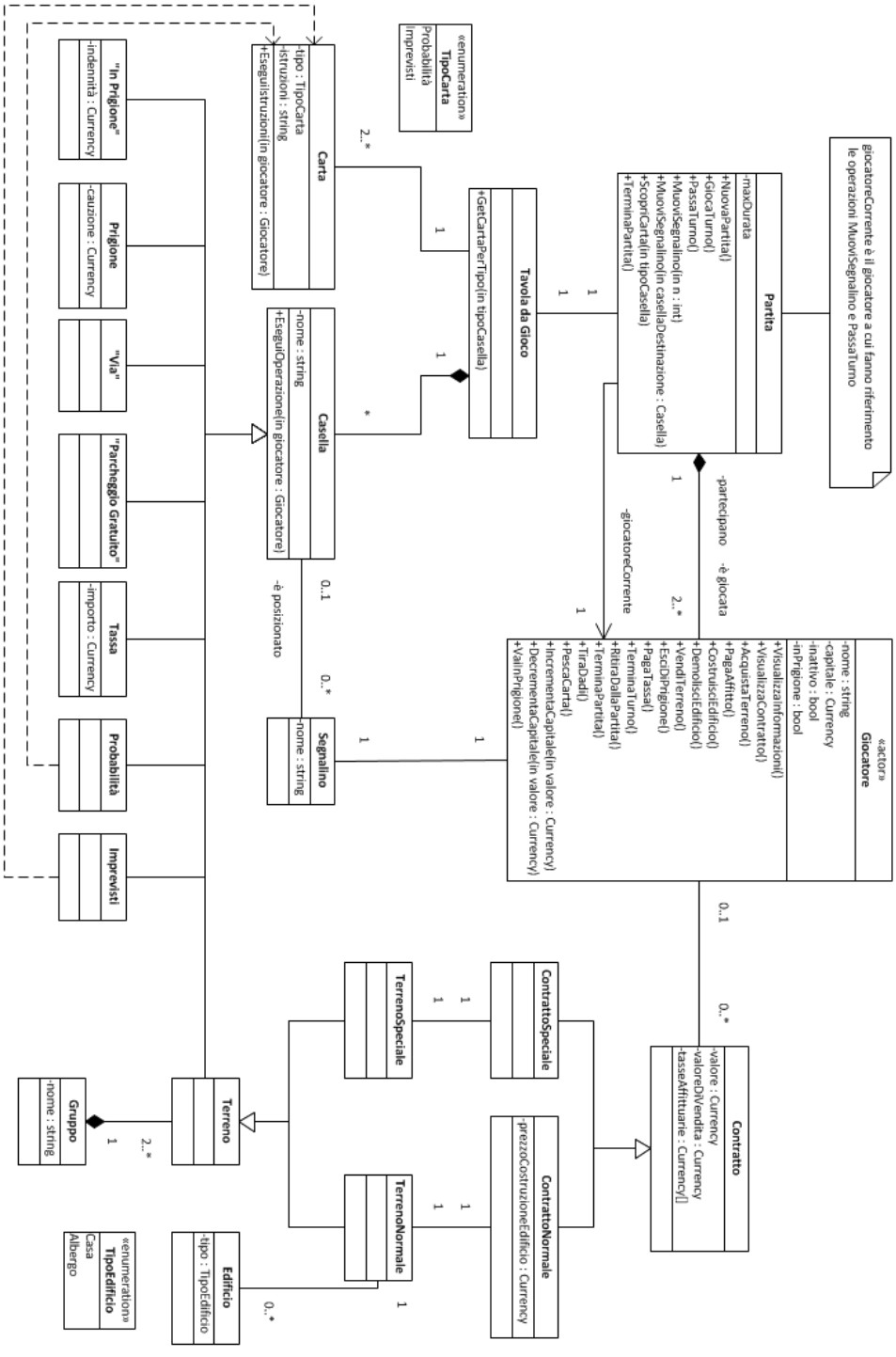
Imprevisti e Probabilità: sono caselle che obbligano a pescare una carta dal mazzo corrispondente e ad eseguirne le istruzioni riportate.

Parcheggio gratuito: è una casella che non comporta nessuna operazione per il giocatore.

Tassa: è la casella che impone il pagamento di una specifica somma di denaro.
Attributo:

- importo: somma da pagare.

DIAGRAMMA COMPLETO CLASSI DI ANALISI



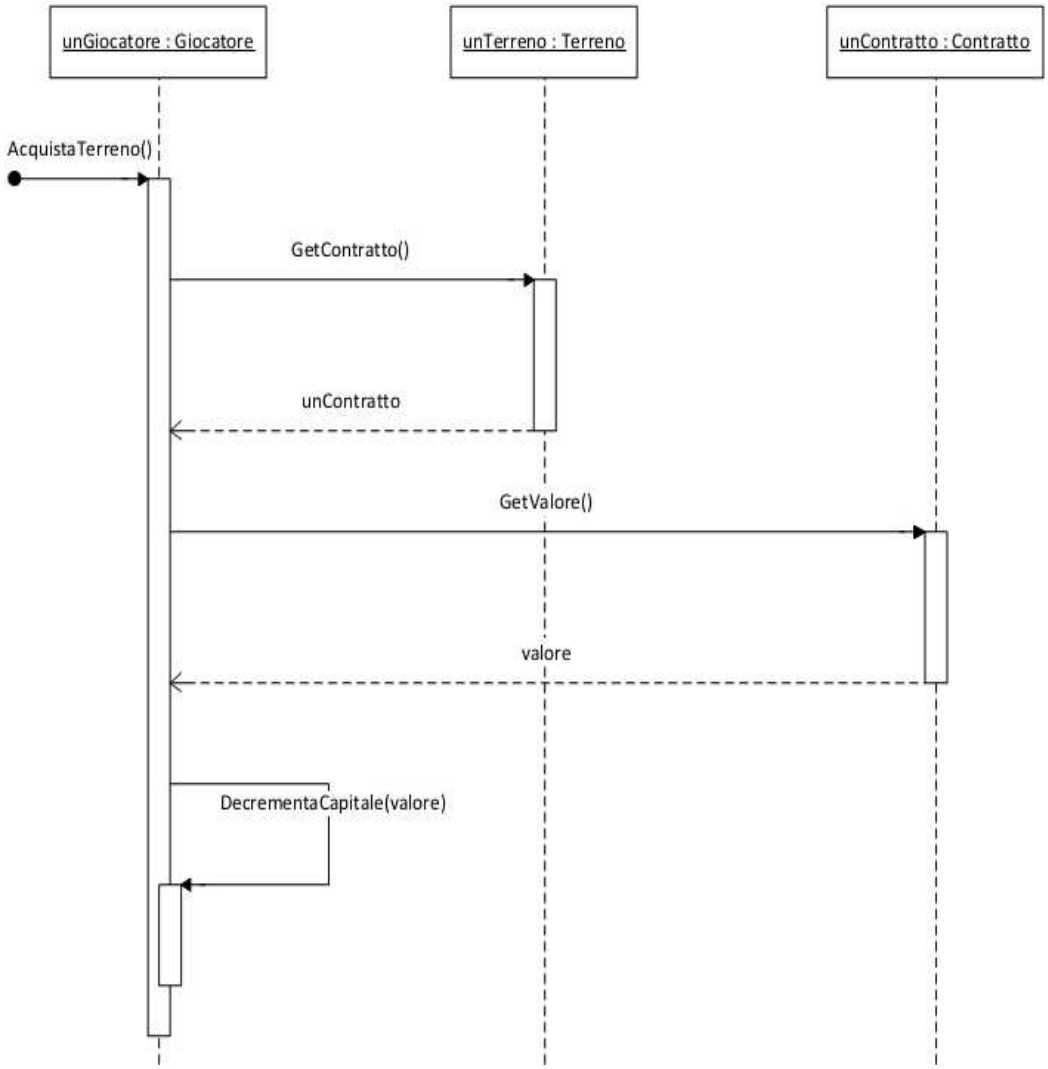
NOTE:

Benchè nella rappresentazione sopra la cardinalità sia stata mantenuta generica (*) per motivi di estendibilità, il Regolamento prevede i seguenti vincoli:

1) I giocatori nella partita sono al massimo 6;
2) In un terreno possono esserci al massimo 4 edifici(cardinalità di edifici per terreno = 0..4); più precisamente:
   - se il proprietario del terreno non possiede tutti i terreni del medesimo gruppo, questa è 0;
   - per gli edifici di tipo Casa è 0..4;
   - per gli edifici di tipo Albergo è 1 se sono già state erette 4 case su quel terreno, 0 in caso contrario.
3) I gruppi di terreni normali contengono 2 o 3 terreni, quello delle società ne contiene 2, quello delle stazioni 4.
4) Le Caselle nella Tavola da Gioco sono in totale 40, di cui 28 Terreni.

DIAGRAMMI DI SEQUENZA

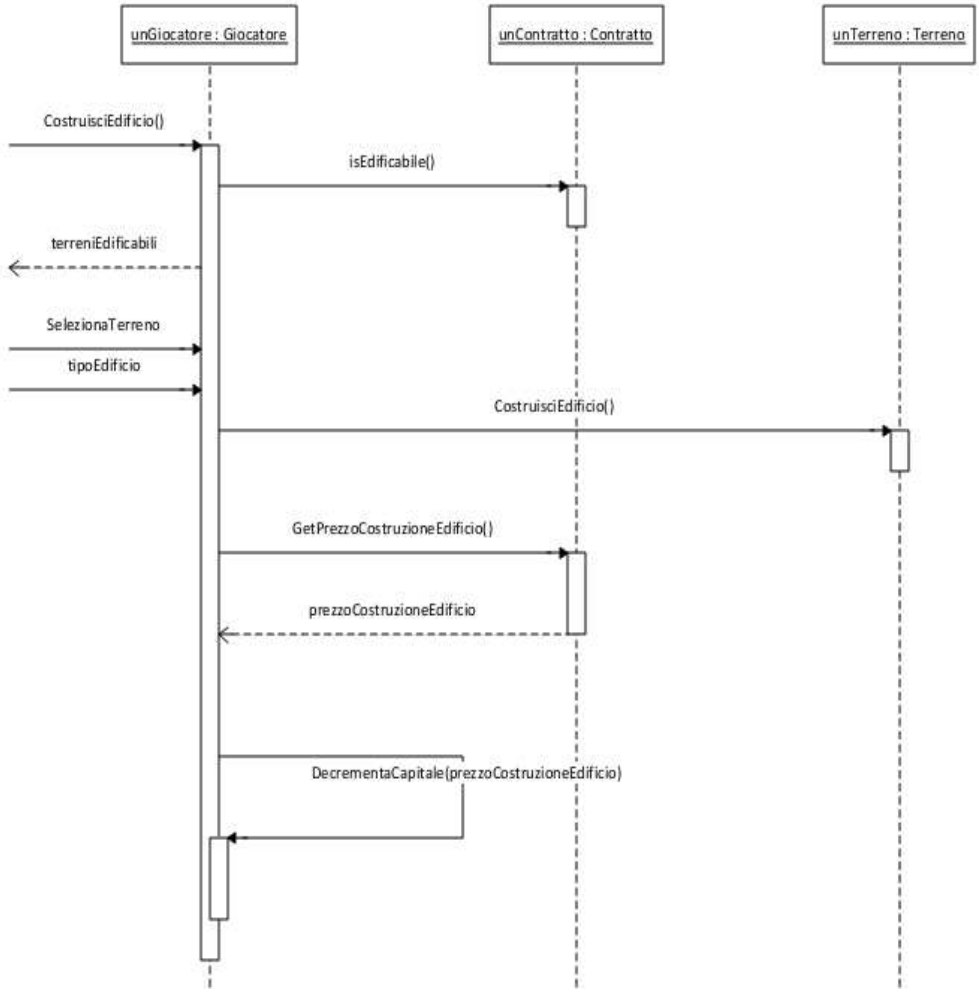
ACQUISTO TERRENO



Quando viene invocato il metodo AcquistaTerreno() sull'entità Giocatore in seguito alla pressione del rispettivo pulsante nell'interfaccia utente da parte di un giocatore, è necessario decrementare il capitale del giocatore del valore del terreno ed associare il contratto del terreno a Giocatore. Il terreno a cui si fa rifermento è quello su cui si trova il segnalino del giocatore, al momento del click sul pulsante Acquista Terreno.
Le operazioni si svolgono nel seguente ordine:

- invocazione del metodo AcquistaTerreno() di Giocatore;
- Giocatore reperisce dal terreno che vuole acquistare il contratto che vi è associato (contenente le informazioni relative al terreno);
- una volta ottenuto il contratto, Giocatore ricava da quest'ultimo il valore di acquisto del terreno che desidera acquistare;
- entrato in possesso di tale valore, Giocatore decrementa il proprio capitale di tale somma.

COSTRUZIONE EDIFICIO

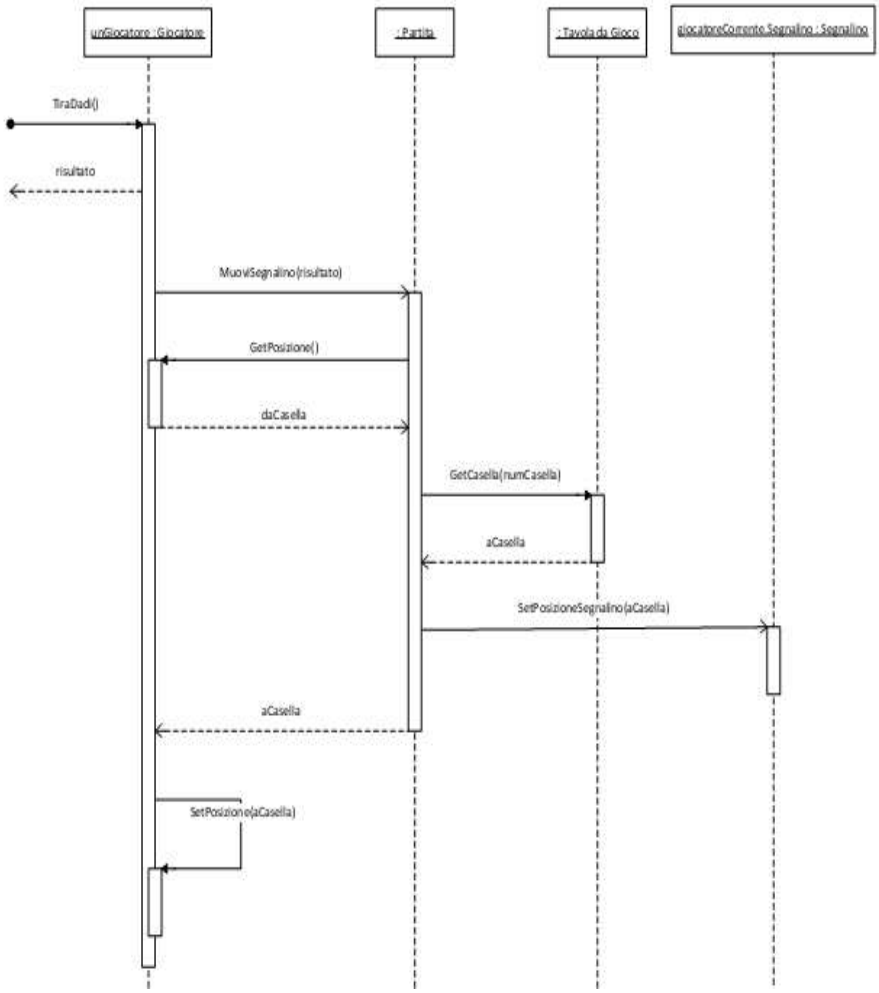


Alla pressione del pulsante Costruisci Edificio nell'interfaccia utente, viene chiamato il metodo CostruisciEdificio() di Giocatore, il cui compito è quello di permettere al chiamante di scegliere, fra i terreni di cui è in possesso e su cui gli è consentito costruire, su quale terreno costruire un edificio, pagando il relativo importo.
Le operazioni si svolgono nel seguente ordine:

- invocazione del metodo CostruisciEdificio() di Giocatore;
- per ogni contratto posseduto dal giocatore, si controlla che il terreno che vi è associato sia edificabile*;
- il sistema presenta al giocatore una schermata contenente tutti i terreni edificabili;
- fra i terreni edificabili il giocatore seleziona quello su cui desidera costruire l'edificio;
- il giocatore ottiene dal contratto associato al terreno selezionato, il prezzo di costruzione di un edificio in loco;
- ottenuto il prezzo di costruzione, Giocatore decrementa i proprio capitale di tale importo.

** un terreno è edificabile, se è classificato come Terreno Normale e se un giocatore dispone di tutti gli altri terreni del gruppo del terreno selezionato. Vedi Glossario voce Edificio.*

TIRO DADI

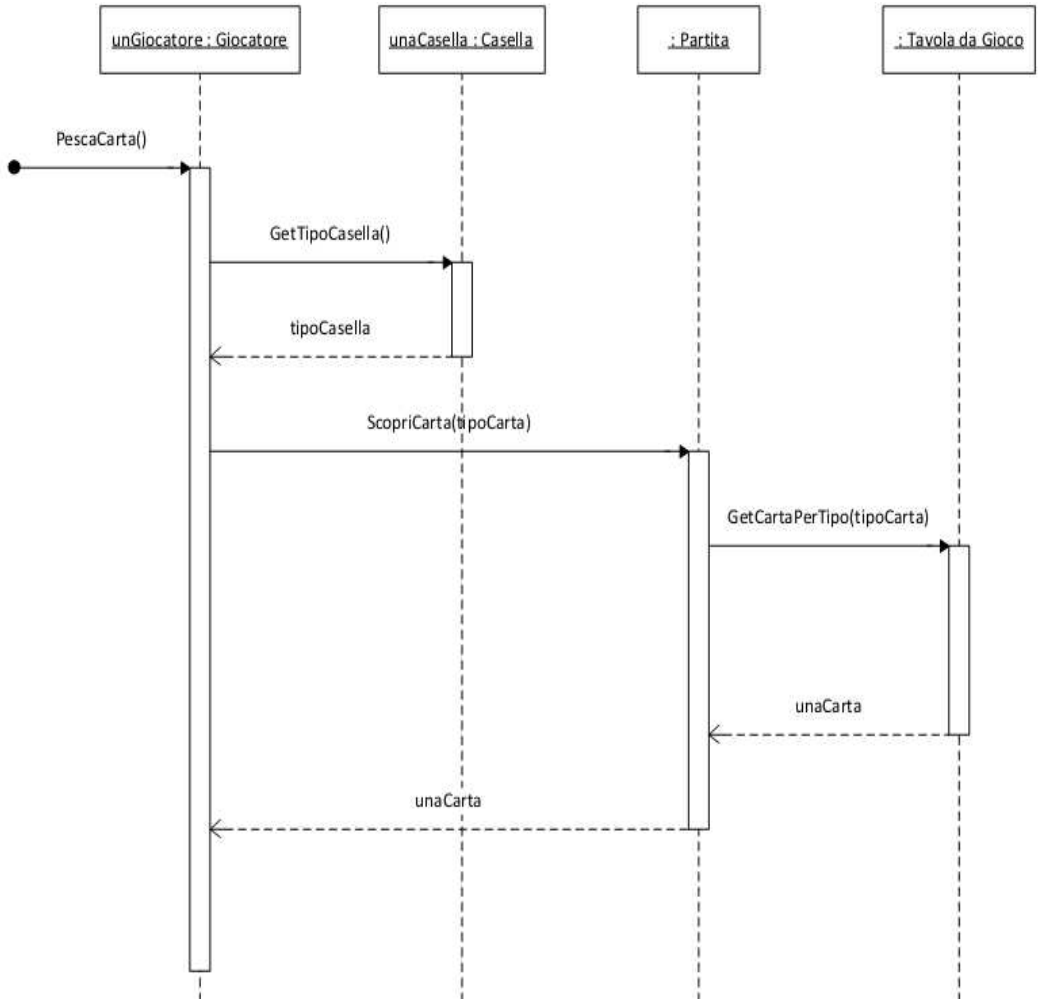


Alla pressione del pulsante Tira Dadi null'interfaccia utente, viene chiamato il metodo TiraDadi() di Giocatore, il cui compito è quello di spostare il segnalino sulla tavola da gioco di un valore casuale compreso fra 2 e 12 (simulazione del lancio di due dadi a 6 facce)

Le operazioni si svolgono nel seguente ordine:

- invocazione del metodo TiraDadi() di Giocatore;
- il valore casuale ottenuto dalla simulazione del lancio dei due dadi a 6 facce è mostrato al giocatore;
- Giocatore esegue il metodo MuoviSegnalino(), passandogli come parametro il risultato del lancio dei dadi;
- MuoviSegnalino ottiene da Giocatore la casella su cui si trova attualmente;
- in base alla casella ottenuta dal giocatore (daCasella) e al valore ottenuto dalla simulazione del lancio dei dadi, Partita ottiene dalla tavola da gioco il valore della casella su cui dovrà collocare il giocatore (aCasella);
- la posizione del segnalino corrispondente a Giocatore corrente in Partita (quello il cui turno è in corso) viene impostata su aCasella;
- la casella di arrivo è ritornata al giocatore come risultato dell'operazione MuoviSegnalino();
- il Giocatore imposta la propria posizione sulla casella di arrivo ricevuta.

PESCA CARTA

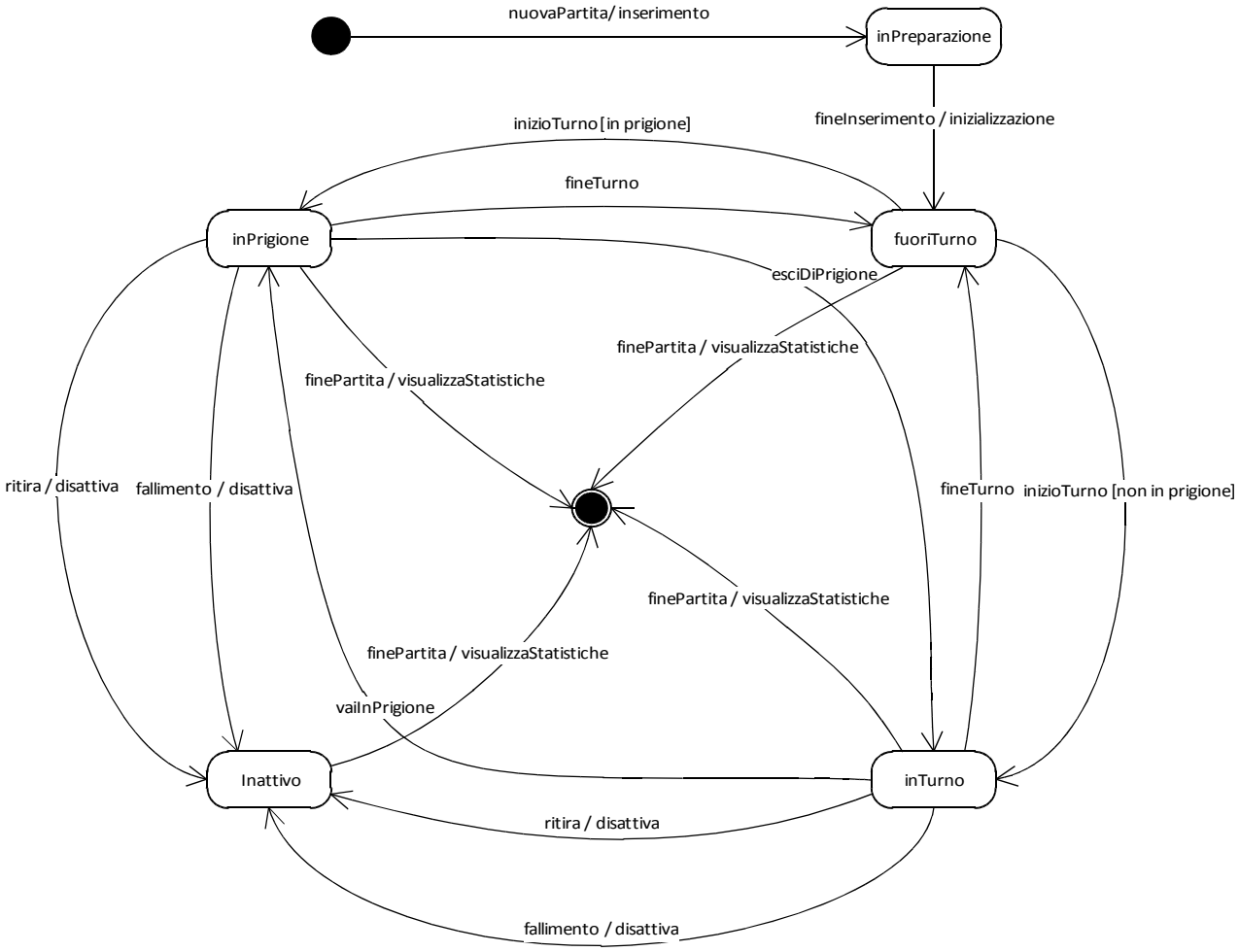


Alla pressione del pulsante Pesca una Carta nell'interfaccia utente, viene chiamato il metodo PescaCarta() di Giocatore, il cui compito è quello di fornire al giocatore la carta del tipo previsto dalla casella su cui si trova (se il giocatore è su una casella imprevisti, egli otterrà una carta di tipo Imprevisti, se si trova su una casella probabilità otterrà una carta di tipo Probabilità).
Le operazioni si svolgono nel seguente ordine:

- esecuzione di PescaCarta() di Giocatore;
- Giocatore richiede alla casella su cui si trova, di fornirgli il proprio tipo;
- in base al tipo ottenuto dalla casella, invoca il metodo ScopriCarta di partita, passando appunto il tipo della casella;
- la partita, sempre basandosi sul tipo della casella ricevuto dal Giocatore, otterrà la carta desiderata da Tavola da gioco;
- partita ritorna a Giocatore la carta appena pescata.

DIAGRAMMI DI STATO

GIOCATORE



Quando l'utente del sistema decide di cominciare una nuova partita, gli viene presentata una form per l'inserimento dei giocatori: Giocatore entra così nello stato di **Preparazione**. Terminato l'inserimento, avviene l'inizializzazione dei giocatori con l'assegnamento di capitale e contratti dopo la quale il Giocatore è attivo e pronto per giocare. Quando diventa il Giocatore **in Turno**, tra le varie situazioni che si possono verificare in questo stato vi sono quelle di fallimento e imprigionamento. Nel primo caso, il Giocatore diventa **Inattivo**. Resterà poi in questo stato fino al termine della partita. Nel secondo caso, ovvero se il Giocatore finisce **in Prigione** accede all'omonimo stato. Sia che sia in Prigione, sia che sia in Turno, quando il Giocatore decide di terminare il proprio turno di gioco, questo torna nello stato **Fuori Turno** dove rimane finchè tutti gli altri giocatori non hanno giocato un turno; poi, ritorna nello stato in cui si trovava prima di terminare il turno. Dallo stato in Prigione, il Giocatore può passare allo stato in Turno uscendo dalla prigione secondo le modalità definite nel regolamento. Il Giocatore in Prigione può anche fallire.
Infine, da ogni stato è possibile terminare la partita.
Terminata la partita, viene presentata una Form con le statistiche della partita, che illustra la classifica dei giocatori e i relativi capitali.

FASE DI PROGETTAZIONE

REALIZZAZIONE DEL SOFTWARE

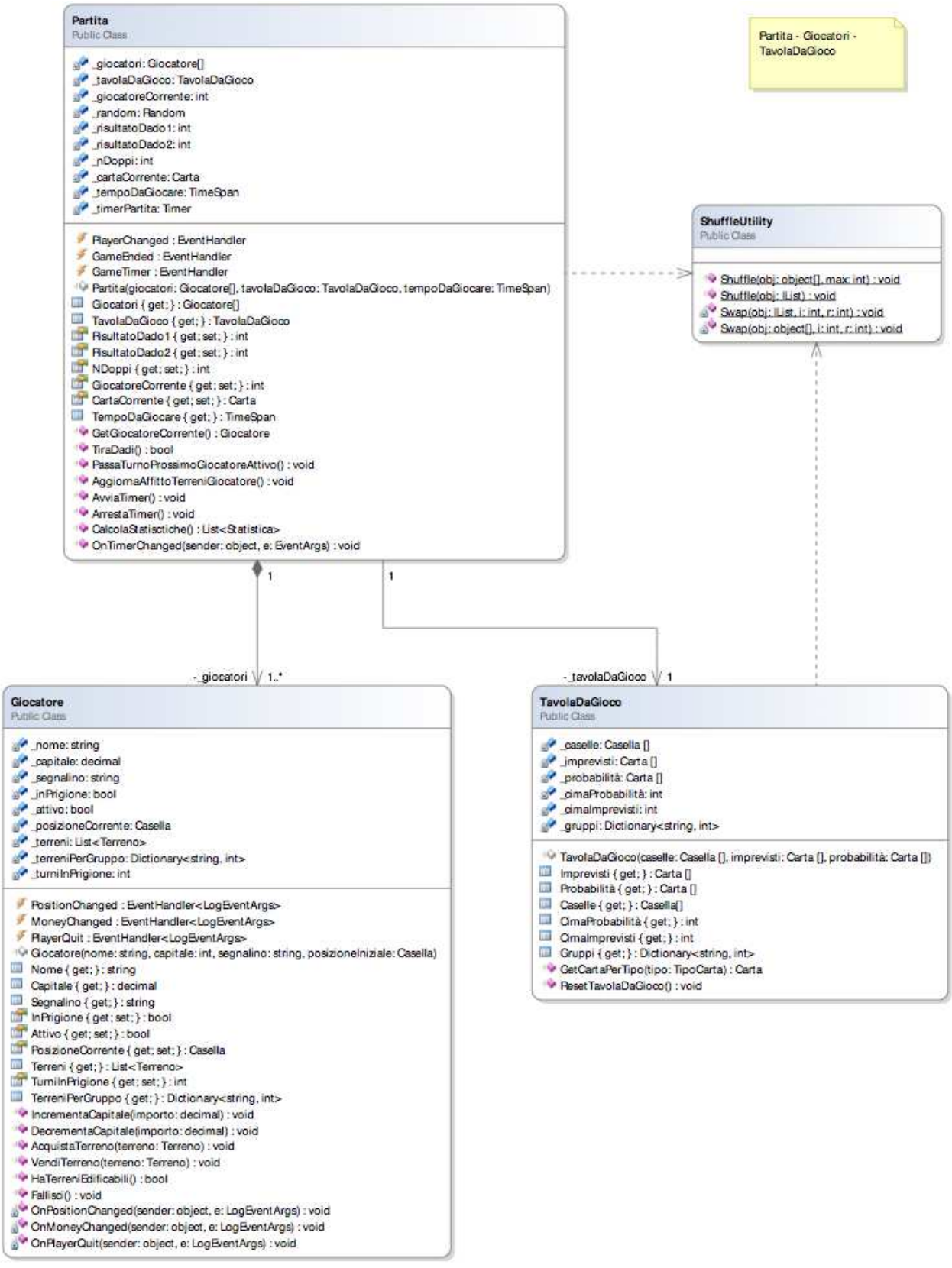
In questa sezione ci occupiamo di presentare il modo in cui il videogioco è stato progettato.
La prima decisione da affrontare è stata quella relativa alle librerie da utilizzare per la realizzazione del software. Avendo il cliente esplicitamente chiesto un videogioco interamente compatibile con tutte le piattaforme Microsoft Windows a partire da Windows XP, la scelta è inevitabilmente ricaduta sul Framework .NET 3.5, con esclusivo utilizzo del linguaggio C#.
Poiché il videogioco da realizzare è stato richiesto in 2D, e non avendo bisogno di animazioni visive particolari, non è stato necessario usufruire di librerie grafiche di terze parti.
Si è deciso pertanto di utilizzare le Windows Forms, interamente integrate nel Framework .NET e che offrono un buon supporto per la creazione di un'interfaccia grafica poligonale come quella del Monopoli.
Durante lo sviluppo dell'applicazione si è tenuto particolarmente conto dei principi di riutilizzabilità ed estendibilità del software, perciò si è considerato necessario riuscire a realizzare un videogioco facilmente modificabile a fronte di aggiunte e/o cambiamenti di specifiche da parte del cliente.
Trattandosi di un progetto relativamente ampio e di una certa complessità, abbiamo ritenuto opportuno suddividere l'applicazione in più moduli durante la progettazione, tentando di renderli autonomi e riducendo le dipendenze da classi concrete ove possibile.
Alcune classi sono dotate di metodi e features non strettamente necessari per l'applicazione specifica, o perlomeno non utilizzati poiché non richiesto nelle specifiche. Questo per permettere di implementare più facilmente nuove funzioni senza dover modificare radicalmente le classi durante il ciclo di vita del software.
Si sono realizzati, procedendo per moduli, dapprima i diagrammi UML delle classi, basandosi sulle classi di analisi, poi si sono implementate le classi in C#.
E' stato dapprima progettato il Model, seguito poi da una View principale e da un Controller che ci permettessero di testare il funzionamento del Model stesso; successivamente sono state a mano a mano implementate tutte le classi di View più complesse, gestite infine da un sub-controller.
Per facilitare la comprensione complessiva del codice, si è provveduto a rendere tutti i metodi il più leggero possibile, suddividendoli in regioni all'interno delle classi, a seconda della funzione di ogni specifico metodo. Ogni classe contiene solamente variabili e metodi necessari per quella classe specifica; le classi che necessitano di funzioni ausiliarie delegano altre classi apposite per lo svolgimento di quelle specifiche funzioni.

E' stato scelto di presentare le classi di progettazione suddivise per moduli di funzionamento, per poi far vedere a mano a mano l'interazione tra i differenti moduli.

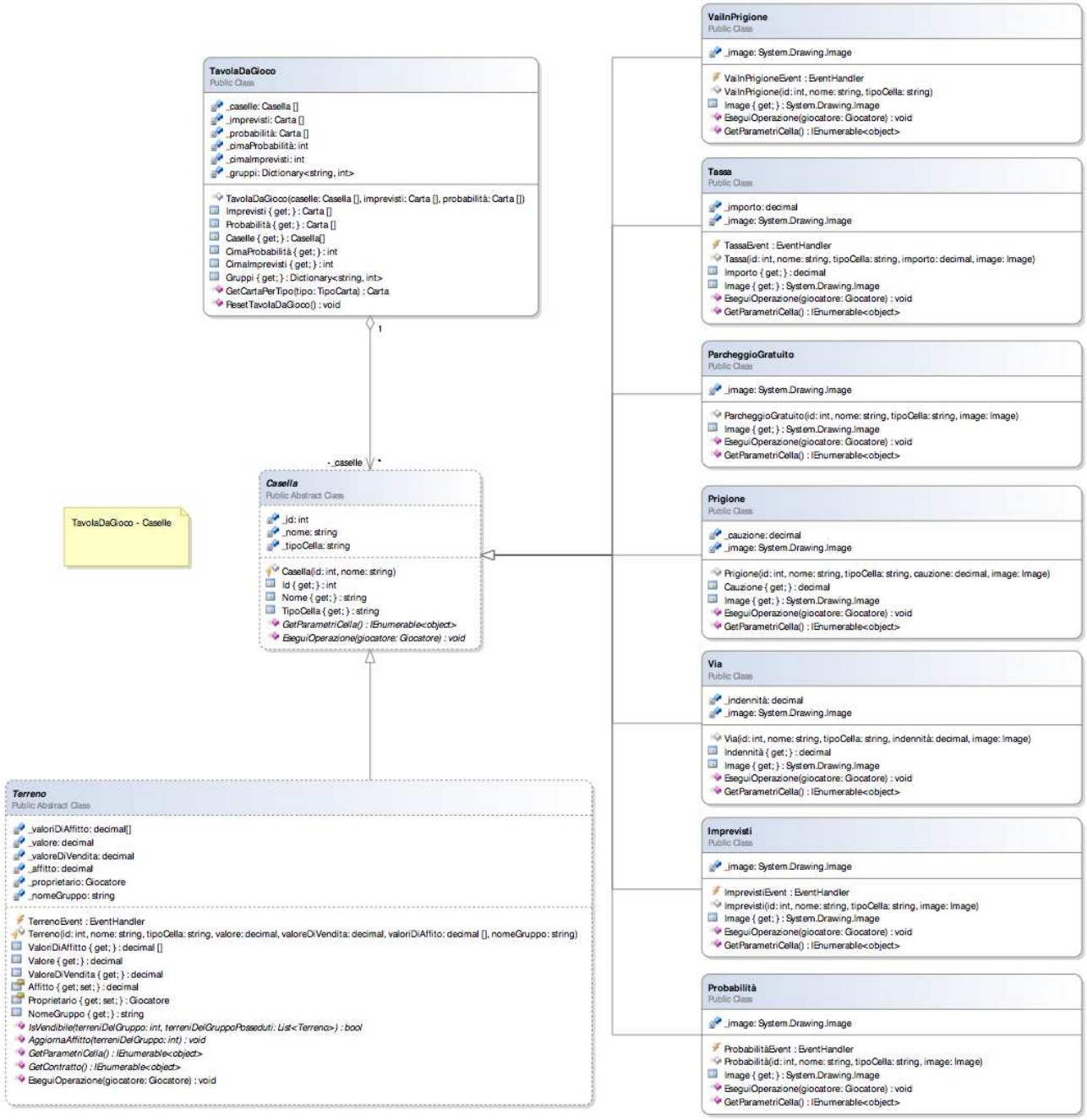
CLASSI DI PROGETTAZIONE

Per la realizzazione dell'applicazione è stato scelto di utilizzare un **pattern MVC** (Model-View-Controller), costruendo un model contenente tutto lo stato e le entità del gioco, e rendendolo il più possibile indipendente dal resto dell'applicazione.

MODEL

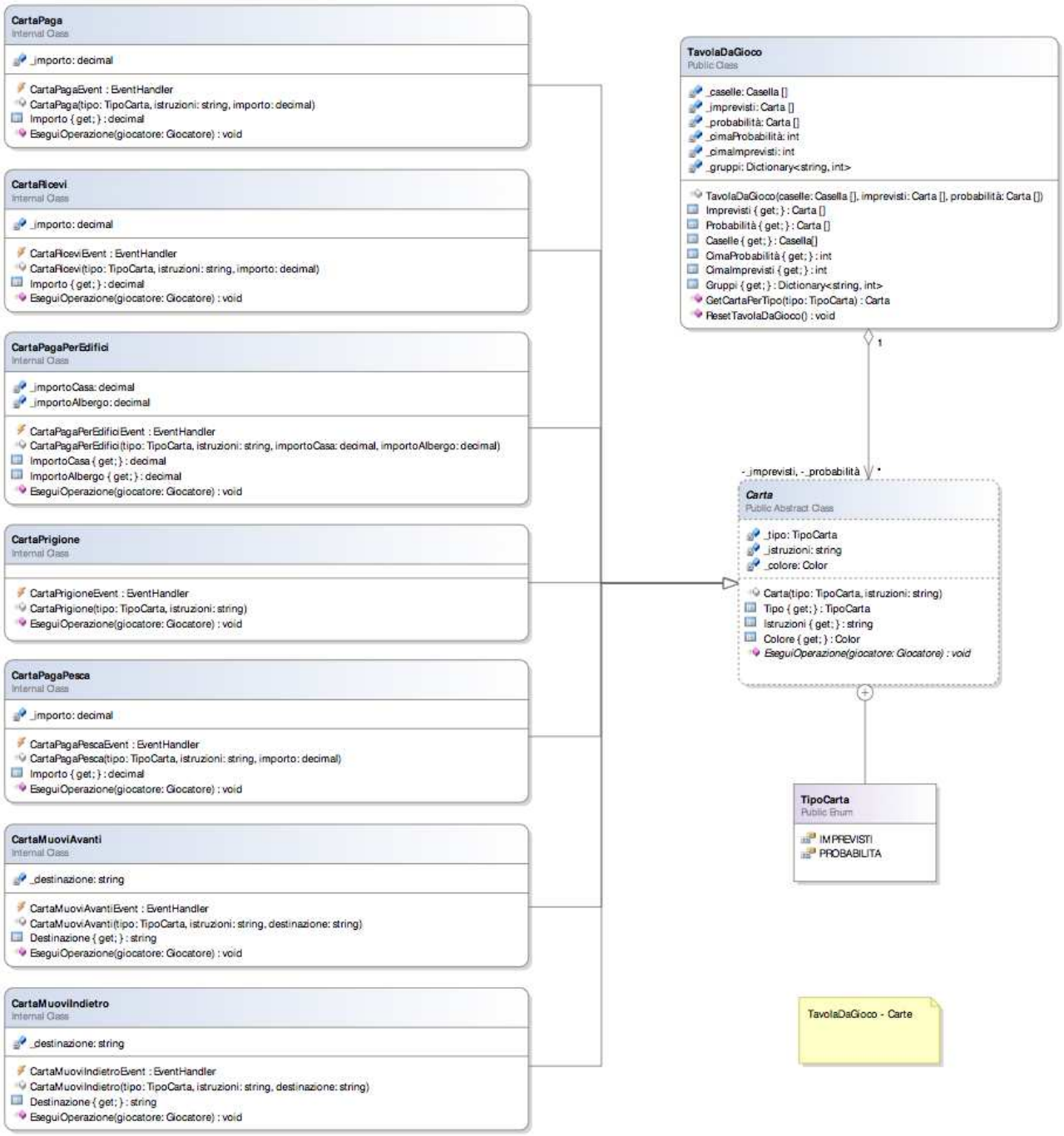


La classe principale del Model rimane **Partita**, che contiene sie le istanze dei **Giocatori** che partecipano alla partita, sia la **Tavola da gioco**, contenente la logica di tutte le **Caselle** e le **Carte** del Monopoli.
Sia Partita sia TavolaDaGioco utilizzano una classe di Utility esterna **ShuffleUtility**, necessaria per mescolare le Carte e i Giocatori (per stabilirne l'ordine di gioco).



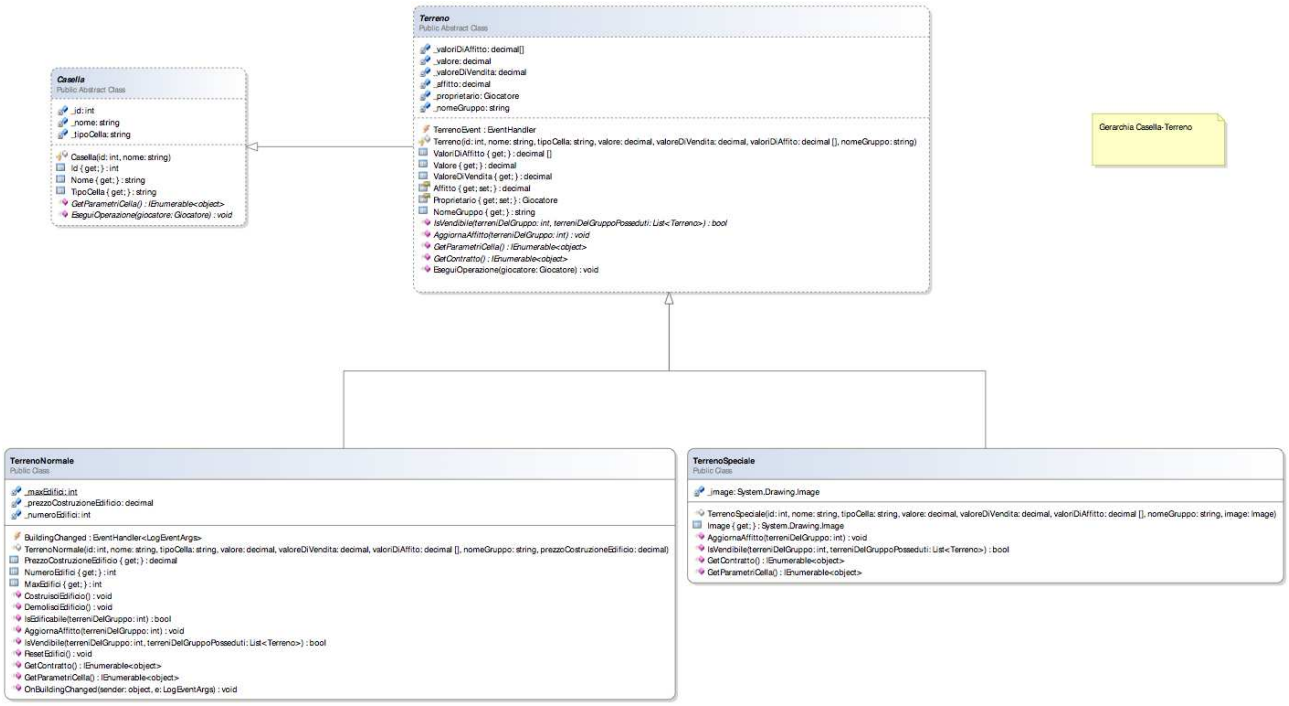
Tutte le Caselle contenute nella Tavola da Gioco, derivano da una **classe astratta Casella**.
Per poter implementare algoritmi e comportamenti differenti nelle varie caselle concrete, viene utilizzato un **Template Method pattern**. In particolare il pattern è necessario per il metodo EseguiOperazione e GetParametriCella, che rispettivamente devono: permettere alle caselle di effettuare operazioni specifiche quando un

Giocatore si ferma su una di esse; restituire in un IEnumerable<object> l'elenco delle informazioni necessarie per la costruzione delle rispettive caselle grafiche.



Vale lo stesso discorso per le Carte, anch'esse contenute nella Tavola da Gioco: tutte le Carte concrete derivano da una classe astratta Carta, e ne implementano il metodo EseguiOperazione, facendo sempre uso del **pattern Template Method**; a differenza delle Caselle, le Carte non necessitano di una rappresentazione grafica particolare, pertanto sono sprovviste del metodo aggiuntivo GetParametri.

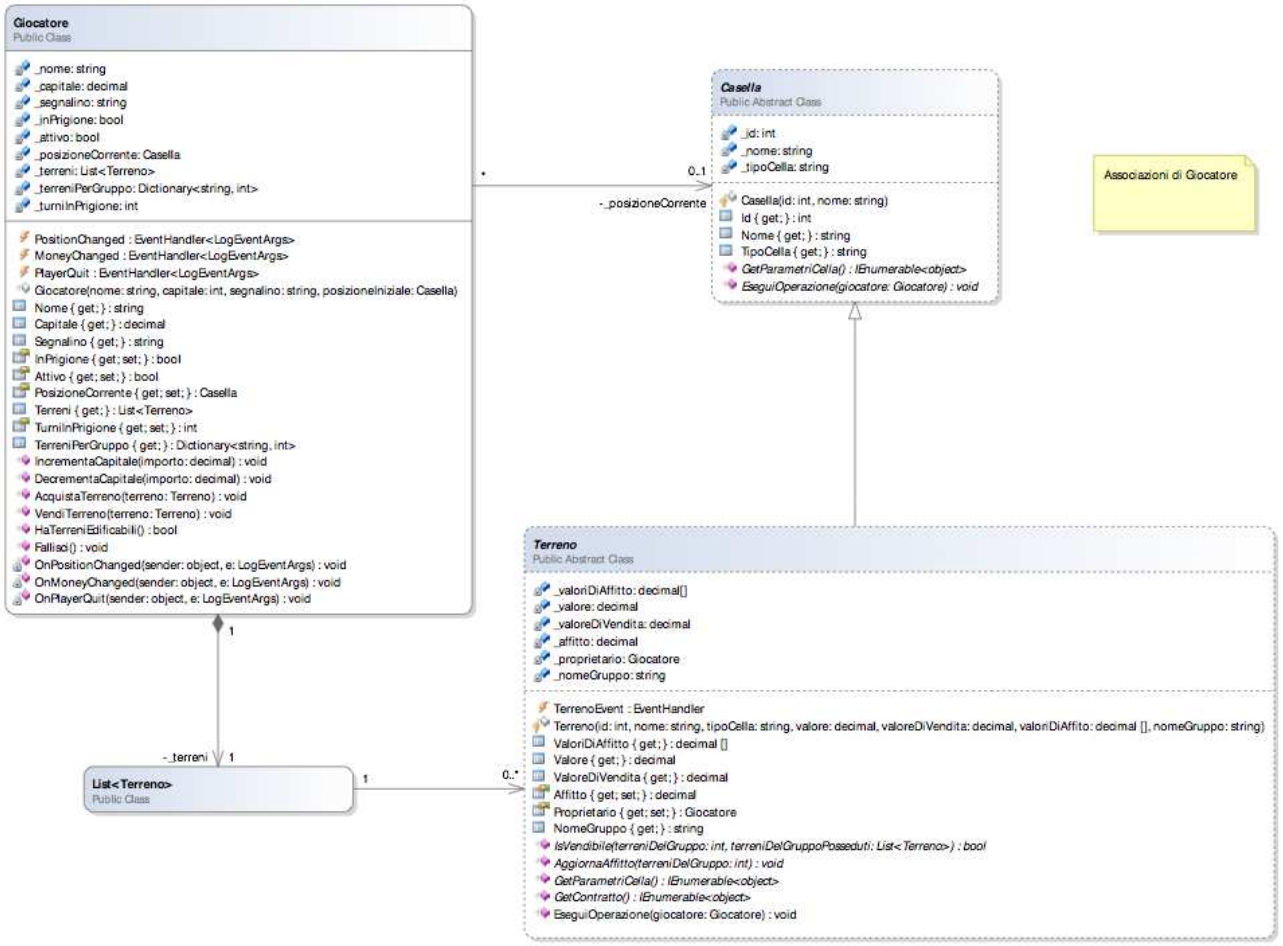
Rispetto alle classi di analisi, è stato scelto di rendere la classe Carta astratta e utilizzare delle sottoclassi in gerarchia per poter ottenere comportamenti diversi a seconda del tipo di Carta, da non confondere con l'enumerativo TipoCarta.



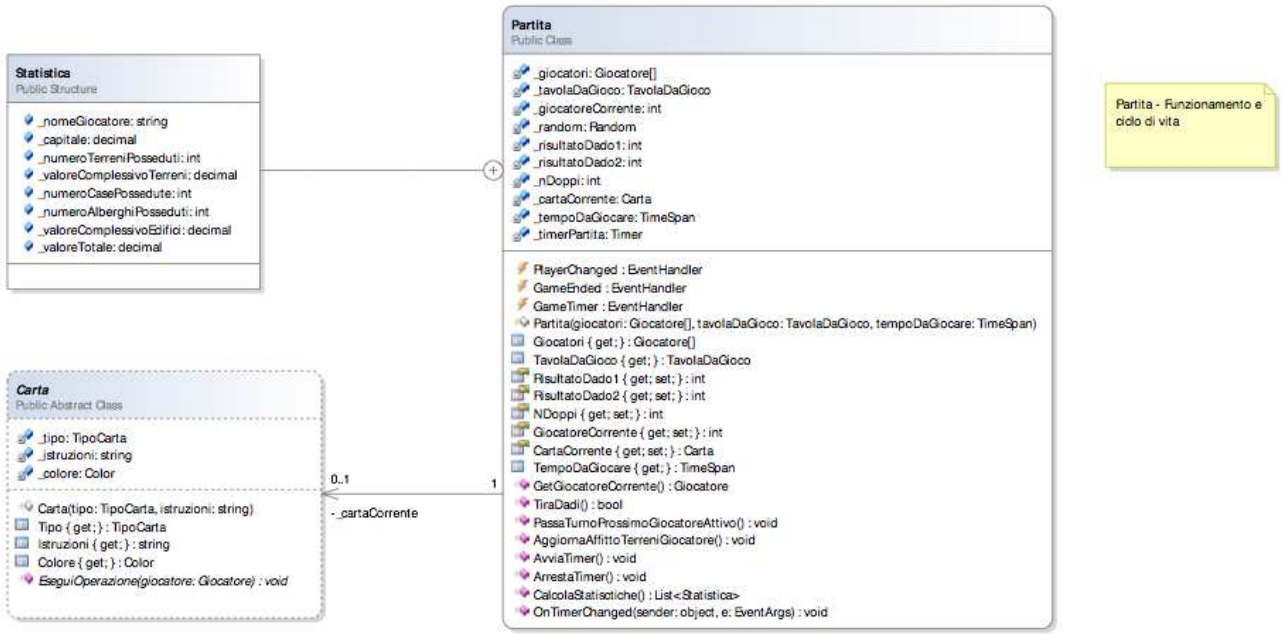
Come nelle classi di analisi, è possibile accorpare in sole due classi i diversi tipi di **Terreni** del Monopoli: le stazioni ferroviarie e le imprese di pubblica utilità sono state raggruppate nella classe **TerrenoSpeciale**, mentre tutti i terreni colorati sono istanze della classe **TerrenoNormale**.

Rispetto alle classi di analisi sono state apportate delle modifiche:

- E' stata rimossa la classe Gruppo, a cui ogni istanza di Terreno era associata (sia normale che speciale), e la si è sostituita con un attributo Gruppo di tipo string, presente all'interno di Terreno.
- E' stata eliminata la classe Contratto, con la relativa gerarchia. Questo perché la presenza di un'ulteriore classe di modello "Contratto" non farebbe che duplicare le informazioni altrimenti già presenti nelle diverse istanze di Terreno. Il cliente avrà tuttavia ancora la possibilità di visualizzare un contratto nell'interfaccia grafica, e le informazioni necessarie verrano reperite dalla classe Terreno stessa, tramite il metodo GetContratto, anch'esso implementato grazie al pattern Template Method.
- La classe Edificio è stata di fatto collassata all'interno della classe TerrenoNormale, alla quale era precedentemente associata. Gli edifici vengono ora gestiti tramite attributi, e la distinzione tra Case ed Alberghi viene fatta in base al numero, come da regolamento.
- L'affitto (o tassa affittuaria) di un Terreno, dipendendo da molteplici fattori durante lo svolgimento di una partita, viene aggiornato dinamicamente grazie al metodo AggiornaAffitto, che reperisce il valore di affitto necessario da un'array di valori di affitto (rentValues), passato al costruttore e vi applica opportuni calcoli, se necessario.



Avendo eliminato la classe Contratto, un Giocatore possiede una lista di Terreni, inizialmente vuota.
Si è optato per rendere l'associazione Giocatore – Casella unidirezionale, e rendere dunque le Caselle ignare dei Giocatori che vi sono posizionati sopra.



La classe di facciata del modello, alla quale sia la View, sia il Controller sono in grado di accedere direttamente è Partita. La classe **Partita** dirige il Model stesso, in quanto contiene il riferimento a tutte le altre classi contenenti uno stato.
Come richiesto dal Cliente, è possibile giocare una partita a tempo; per questa ragione Partita contiene un attributo di tipo Timer e due metodi opportuni AvviaTimer e ArrestaTimer, che verranno utilizzati in caso l'Utente specifichi un tempo di gioco non nullo. Per semplicità si è preferito lavorare con valori temporali del minuto, impedendo quindi all'utente di specificare i secondi nella durata massima di una partita.
Partita contiene inoltre un riferimento alla Carta corrente, ovvero a una carta pescata da un Giocatore, che però non ne ha ancora eseguito le istruzioni. Questo per ridurre al minimo le chiamate che Partita dovrà effettuare a runtime, a fronte di richieste.

La **struttura Statistica** innestata all'interno di Partita è necessaria per il calcolo delle statistiche di fine partita, come richiesto dal cliente. Si è scelto infatti di non far effettuare il calcolo delle statistiche alla stessa finestra di dialogo che le visualizzerà, ma al model, che pertanto deve potersi appoggiare a una struttura in cui mantenere i dati calcolati.

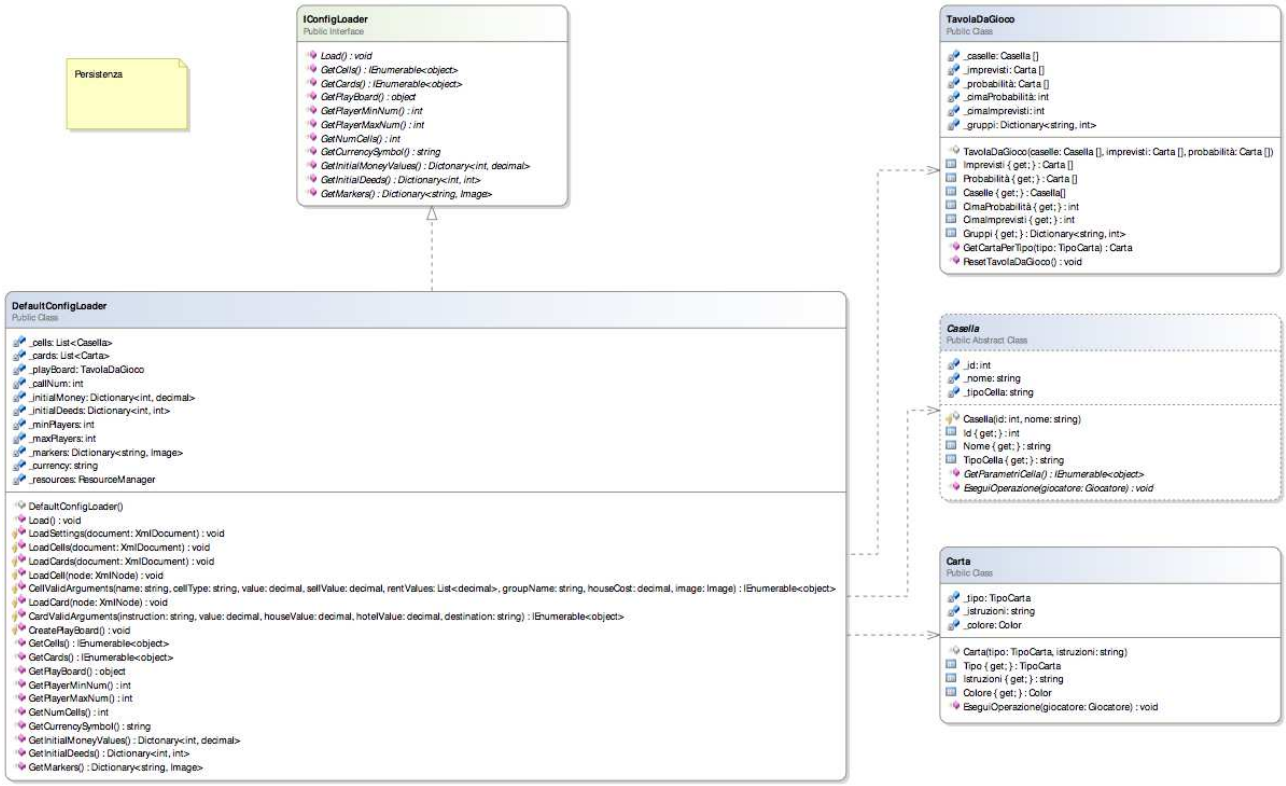
Tutte le proprietà del modello di cui non deve essere possibile modificare il valore a runtime sono state rese Read-Only.

Avendo deciso di adottare un **pattern MVC**, ci siamo affidati a eventi e delegati per poter notificare alla View i cambiamenti di stato del Model. Partita, in particolare, deve poter notificare la View:

- ogni qualvolta vi sia un cambio di turno tra i Giocatori
- ogni volta che scatta il Timer (di minuto in minuto)
- se la partita è giunta al termine, ovvero se si sono presentate le condizioni necessarie per concludere la partita (è scaduto il tempo di gioco, è rimasto un solo Giocatore attivo)

Tratteremo in maniera più approfondita gli eventi lanciati dal **modello** e i relativi delegati dopo aver parlato delle **View** e dei **Controller**.

STRATEGIE DI PERSISTENZA



Anziché istanziare tutte le parti dell'applicazione staticamente, ovvero cablando i parametri da passare ai costruttori nel codice, si è optato per il caricamento dinamico da file dei dati necessari alla configurazione e all'avvio del videogioco.
In particolare si è deciso di unire la fase di caricamento dei dati persistenti alla fase di creazione del modello: è stato dunque adottato un **Builder pattern**, che abbiamo tuttavia deciso di realizzare non mediante una classe astratta, bensì un'interfaccia.
L'interfaccia **IConfigLoader** fornisce infatti tutti i metodi necessari per la costruzione e il reperimento di una tavola da gioco funzionante, oltre che a parametri di configurazione del gioco stesso.
Tutti i builder concreti che implementeranno quest'interfaccia daranno direttamente la possibilità di caricare (metodo Load) i dati necessari per la creazione del modello (in particolare per Carte, Contratti, inizializzazione Giocatori), a seconda della strategia di persistenza. Una volta caricati, e costruite tutte le classi secondo le specifiche, sarà possibile, sempre tramite l'interfaccia, ottenere le classi appena istanziate, oltre a valori come il numero minimo e massimo dei Giocatori, il numero di caselle totali, e altro.

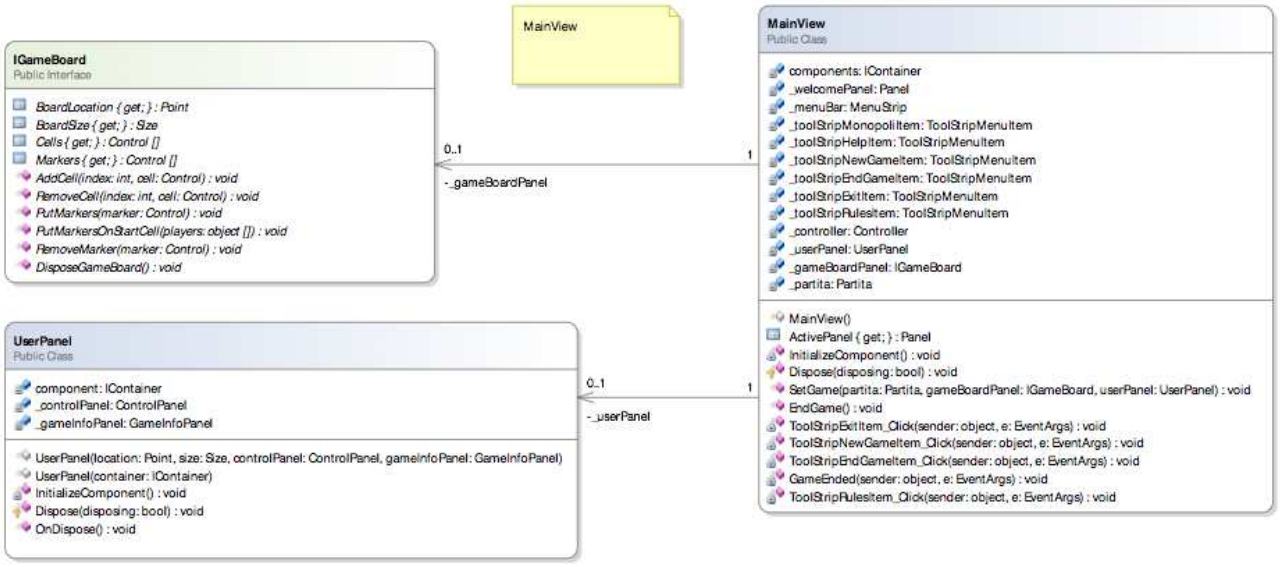
Sebbene l'interfaccia sia stata progettata tenendo conto dello specifico gioco che si andava a realizzare, questa è comunque potenzialmente riutilizzabile per una vasta gamma di giochi da tavolo, che comprendano una tavola da gioco con delle caselle e/o delle carte. I metodi che permettono di ottenere i risultati della creazione del Builder ritornano infatti degli IEnumerable<object>, dunque possono ritornare qualsiasi tipo di collezione di oggetti concreti.

Nello specifico, il Builder concreto **DefaultConfigLoader** accede a un file XML contenente tutte le informazioni necessarie per il Load. Il file XML viene associato ad un **XmlDocument**, che verrà utilizzato nei metodi protected LoadSettings, LoadCells, LoadCards, effettuando le corrispondenti operazioni di Build.

- L'istanziazione di tutte le diverse classi del Model avviene tramite Reflection. Il builder provvede a reperire i dati necessari alla creazione di ogni singolo oggetto dal file XML, passandoli poi al costruttore relativo come array di object.

VIEW

Dovendo realizzare un videogioco è necessario discutere delle scelte progettuali per la costruzione dell'interfaccia grafica.
Volendo creare un gioco facilmente estendibile, è stato deciso di non affidarsi interamente a un'immagine di sfondo contenente la tavola da gioco, in quanto per nulla modificabile. Abbiamo dunque trovato una soluzione che ci permettesse di creare a runtime sia le caselle grafiche, sia il resto dell'interfaccia utente.
Benché la creazione richieda molte più classi e moduli di quanto non avrebbe fatto il caricamento statico dell'immagine della tavola da gioco completa, a fronte di cambiamenti nei requisti del cliente, garantische la possibilità di effettuare modifiche senza dover riscrivere grandi quantità di codice e soprattutto senza dover creare nuove immagini o modificare lo sfondo esistente con ambienti di sviluppo e programmi esterni.

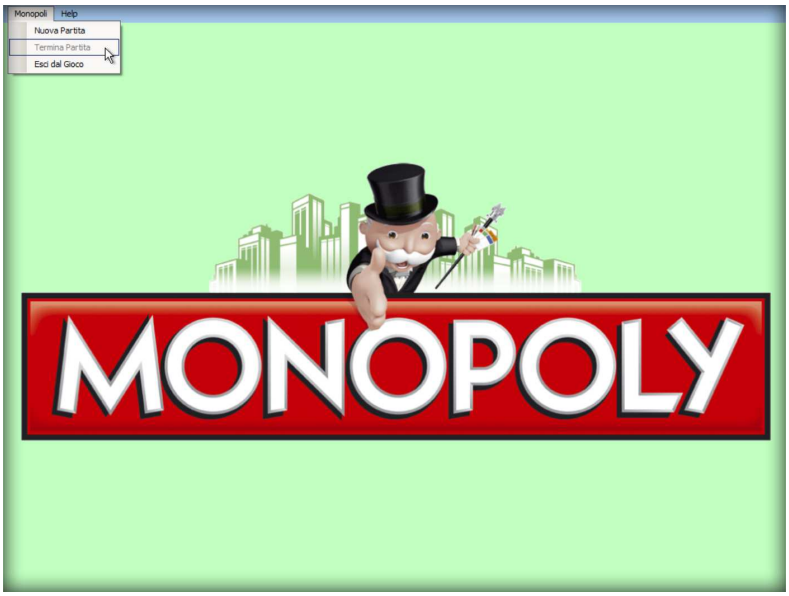


La **MainView** è costituita da una **Form** a tutto schermo, non ridimensionabile, contenente una **Toolbar** e un unico pannello che appare all'avvio dell'applicazione. Quando viene creata una Nuova Partita, il pannello iniziale viene sostituito con altri due Pannelli:

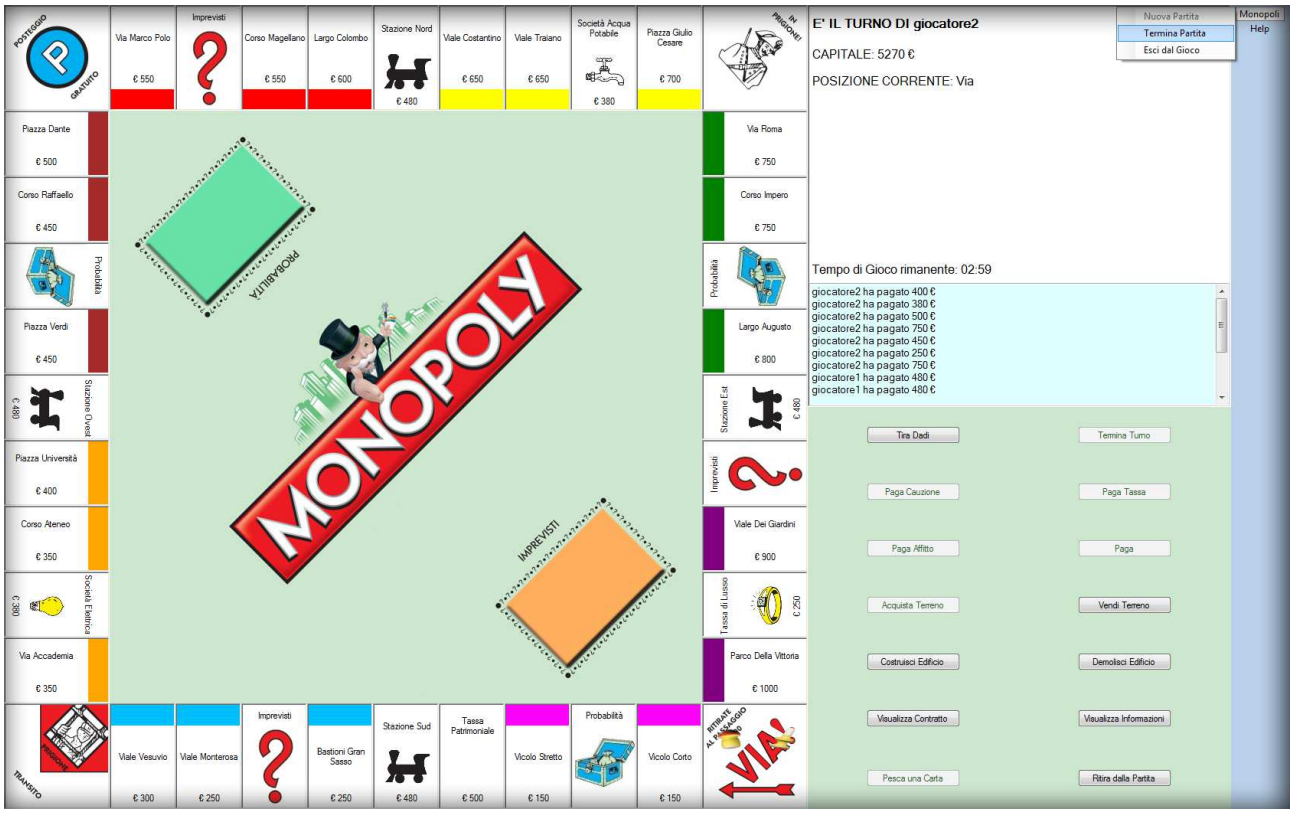
- Un pannello che implementa l'interfaccia **IGameBoard,** contenente la tavola da gioco grafica, assieme ai segnalini dei giocatori;

- Uno **UserPanel**, che funge da Wrapper per altri due pannelli, il **ControlPanel** e il **GameInfoPanel**.

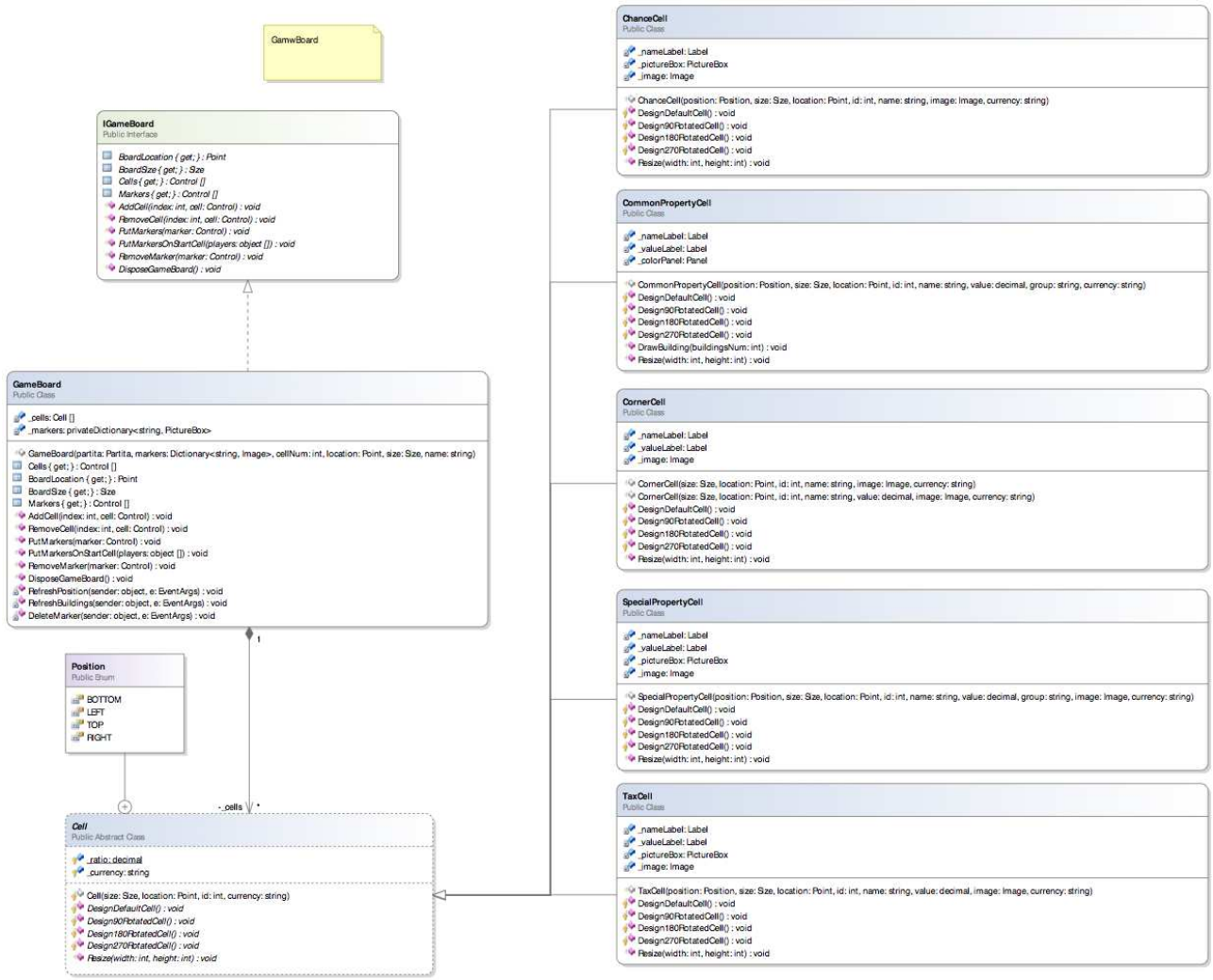
Il wrapper UserPanel non sarebbe strettamente necessario, ma l'abbiamo introdotto per motivi grafici. Infatti la ToolBar contiene alcune funzioni che il cliente ha esplicitamente richiesto essere sempre disponibili. Non volendo togliere spazio sulla parte di schermo dedicato alla **GameBoard**, è stato necessario inserire la toolbar dentro a un altro controllo durante lo svolgimento di una partita: per evitare che la toolbar venisse divisa tra ControlPanel e GameInfoPanel, si è deciso di inserirla all'interno del pannello wrapper, che contiene inoltre i due pannelli appena citati.



Schermata principale della MainView



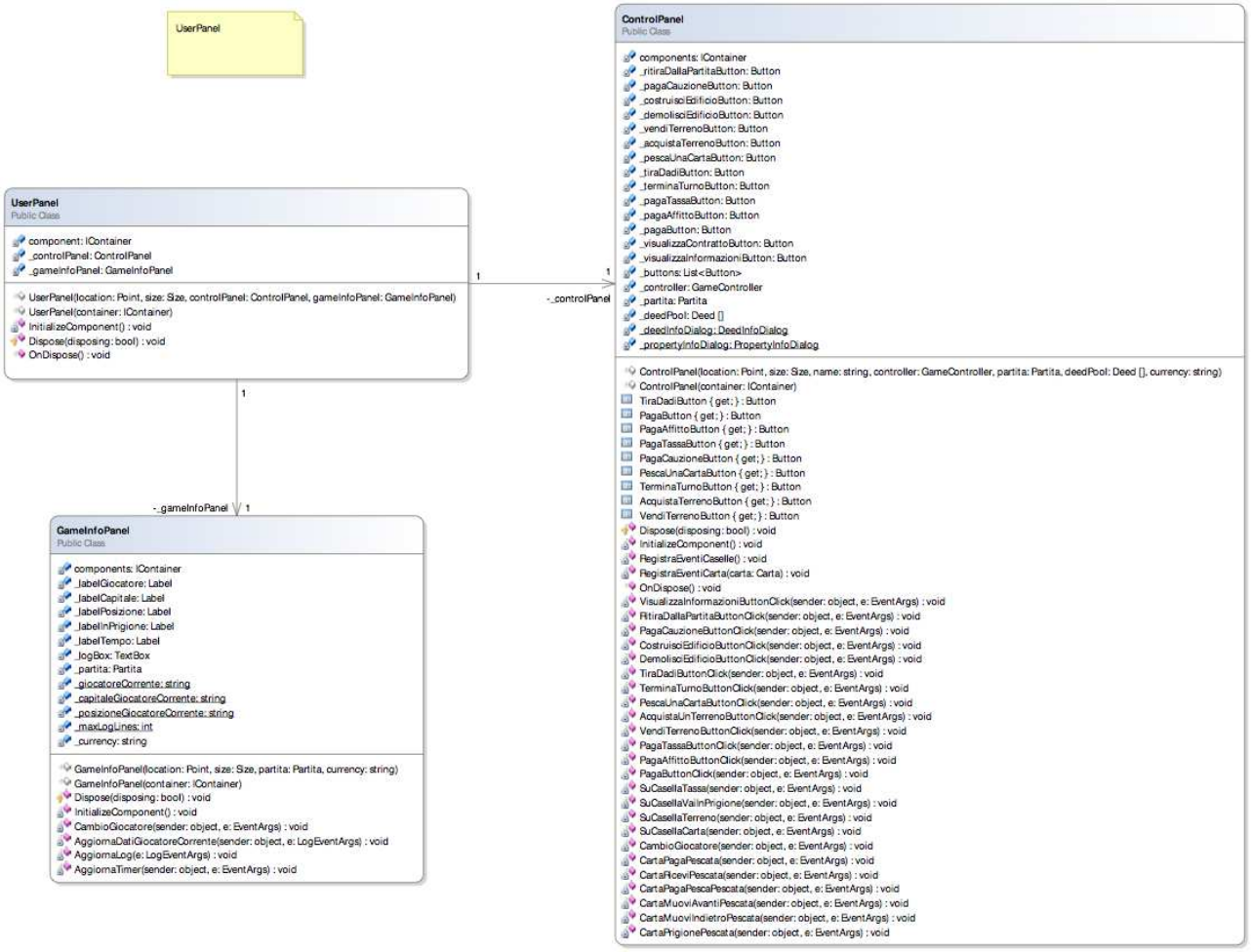
Interfaccia di gioco: a sinistra la GameBoard, a destra lo UserPanel, composto da MenuToolbar, GameInfoPanel e ControlPanel



Si è optato per l'utilizzo di un'interfaccia generica IGameBoard, contenente i metodi base e necessari per qualsiasi tipo di gioco da tavolo. In questo modo può essere ampiamente riutilizzata in progetti simili futuri.

La classe che implementa IGameBoard è **GameBoard**, che estende inoltre la classe Panel di Windows Forms, ed è associata ad un **array di Cell**.
La classe astratta Cell eredita da Panel e rappresenta le singole caselle grafiche. A seconda della rappresentazione grafica delle singole caselle, sono state implementate differenti caselle grafiche concrete. La costruzione delle singole Cell concrete avviene grazie a **Template Method pattern**: a seconda del valore dell'enumerativo **Position** passato tramite costruttore, una Cell si inizializza chiamando l'opportuno metodo protected, a seconda che debba essere ruotata di 0, 90, 180 o 270 gradi.
Si è presupposto per semplicità che le caselle grafiche del Monopoli potessero solamente essere ruotate dei valori sopra elencati; questo inoltre per via dei limiti imposti dalla stessa classe Panel del Framework. Qualora ci fosse necessità di ruotare un pannello di un angolo differente, sarebbe necessario sviluppare una nuova funzione apposita.

E' stato in più inserito in Cell un metodo astratto pubblico di Resize, non implementato nelle classi concrete. Questo per fornire uno scheletro di base dentro il quale poter realizzare un algoritmo per il ridimensionamento delle celle grafiche stesse, e quindi della finestra, non contemplato in questo prototipo.



Come già accennato, lo **UserPanel** funge da wrapper di due altri componenti: il **GameInfoPanel** e il **ControlPanel**, che assieme alla **GameBoard** costituiscono il cuore dell'interfaccia grafica del videogioco.

Durante lo sviluppo del software e delle meccaniche di gioco ci siamo resi conto che, per testare il corretto funzionamento dei moduli, era fondamentale avere un'interfaccia grafica su cui appoggiarsi, anche se rudimentale.
Se da un lato la GameBoard permetteva di visualizzare subito la tavola da gioco, in maniera da verificarne il corretto caricamento, dall'altro c'era bisogno di un pannello che permettesse di visualizzare le operazioni che venivano effettuate e come venivano effettuate. Trattandosi di un videogioco, abbiamo escluso a priori la possibilità di fare Testing tramite standard output su console, e abbiamo invece creato il prima possibile tutte le finestre di dialogo e tutti i pannelli grafici su cui visualizzare direttamente i risultati delle operazioni.

Avendo già una GameBoard funzionante, era dunque fondamentale costruire dei controlli che permettessero l'interazione con l'utente finale. Per interagire con l'utente si è optato per la soluzione più semplice e intuitiva: una batteria di **pulsanti grafici Button**, associata a delle Label di testo che contenessero tutte le informazioni necessarie all'utente per seguire l'andamento del gioco.
Inizialmente si era ideato un pannello che contenesse entrambi i componenti sopra elencati. A mano a mano che si sono implementate nuove funzionalità di gioco ci siamo tuttavia resi conto che il pannello diventava sempre più una Fat Class, carica di eccessive responsabilità; si è pertanto deciso di suddividere il pannello iniziale in due classi distinte (raggruppate poi dentro allo UserPanel), in modo da suddividere equamente le responsabilità:

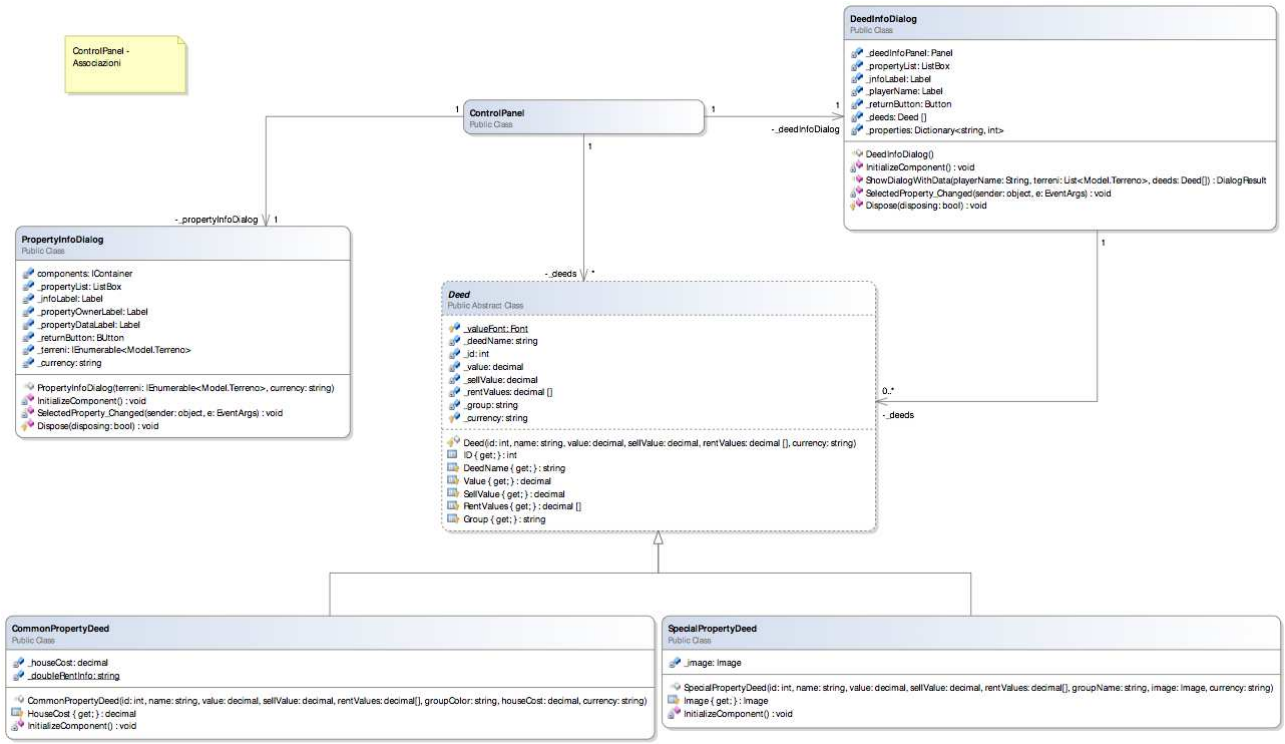
- Il **GameInfoPanel** è rimasto il pannello grafico contenente solamente le informazioni sull'andamento del gioco, dei turni e dello stato del Giocatore corrente, rispettivamente tramite una TextBox di log, e svariate Label contenenti informazioni specifiche.
- Il **ControlPanel** detiene invece il controllo dei pulsanti grafici con cui può interagire l'utente.

Il GameInfoPanel ha come unica mansione quella di aggiornarsi seguendo il pattern Observer, caratteristico del MVC, rispondendo dunque a modifiche di stato nel Model. Non deve compiere operazioni attivamente nè ha il compito di chiamare procedure su altre classi, ed è di fatto un modulo completamente a sé stante. In particolare il Log aggiunto in questo pannello si è rivelato estremamente utile in fase di Testing.

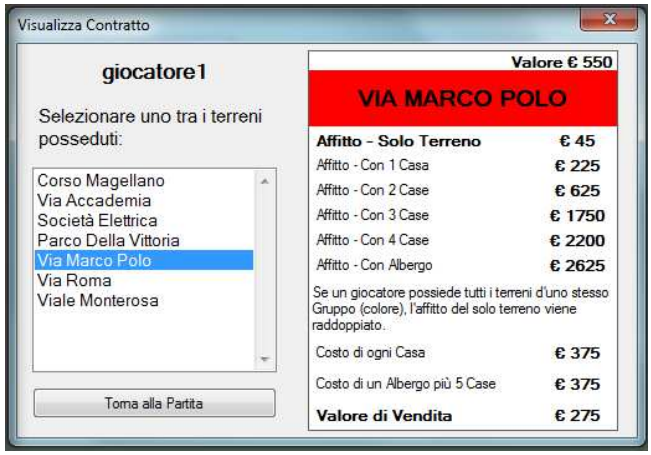
Seguendo gli schemi e le descrizioni dei casi d'uso, si è realizzato un pulsante grafico per ogni funzionalità offerta all'utente; ognuno di questi pulsanti è stato poi collegato a un gestore del click.
Il ControlPanel è pertanto la classe di View che raccoglie tutti gli input dell'utente ed effettua le chiamate corrispondenti.

ControlPanel

Ricevendo tutte le richieste dell'utente, il ControlPanel deve, a seconda del caso, poter invocare un opportuno metodo sul Controller di gioco, che tratteremo a breve, oppure eseguire determinate operazioni grafiche, come la disabilitazione di un pulsante, oppure l'apertura di una finestra di dialogo in aiuto all'utente.
In particolare il ControlPanel è associato a due finestre di dialogo personalizzate statiche: **DeedInfoDialog** e **PropertyInfoDialog**.
Per visualizzare queste due finestre di dialogo non è necessario effettuare calcoli specifici, pertanto la View è in grado di reperire i dati necessari dal Model e mostrarli all'interno di queste finestre.



La DeedInfoDialog è responsabile per la visualizzazione dei contratti grafici. Mentre in analisi i contratti erano trattati come entità di modello, in progettazione si è preferito considerarli solamente delle entità grafiche.
Tutti i contratti derivano dalla classe astratta **Deed**, che eredita da Panel; pertanto ogni contratto non è altro che un pannello che verrà popolato con le informazioni ottenute dalla rispettiva Casella di Model (grazie al metodo GetContratto).



DeedInfoDialog

La finestra di dialogo DeedInfoDialog è quella, nella maggior parte dei casi, più utilizzata durante lo svolgimento di una partita, in quanto permette all'utente di visualizzare i contratti relativi ai terreni posseduti. Si tratta inoltre della finestra di dialogo con i contenuti più "pesanti" da caricare, dovendo costruire a runtime dei pannelli, contenenti anche immagini, di cui è necessario fare il rendering delle dimensioni. A differenze delle celle grafiche, i contratti non sono sempre visibili all'utente, ma solamente quando viene aperta questa finestra di dialogo. Si è cionondimeno preferito adottare un **Object Pool pattern,** creando tutti i contratti grafici una sola volta alla creazione della partita. Questi contratti vengono mantenuti in memoria dal ControlPanel stesso, che può perciò accedervi in qualsiasi istante, quando l'utente chiede di visualizzarli.

Questa scelta è stata fatta per evitare di dover costruire ex novo tutti i pannelli dei contratti ogni volta che viene richiesto di aprire la DeedInfoDialog. La decisione è stata dunque presa dopo aver analizzato possibili problemi prestazionali, volendo evitarli a priori.

A fronte di richiesta della MainView, lo **UserPanel** è in grado di invocare il metodo OnDispose su entrambi i pannelli appena descritti, che a loro volta libereranno tutte le risorse occupate dai controlli contenuti.

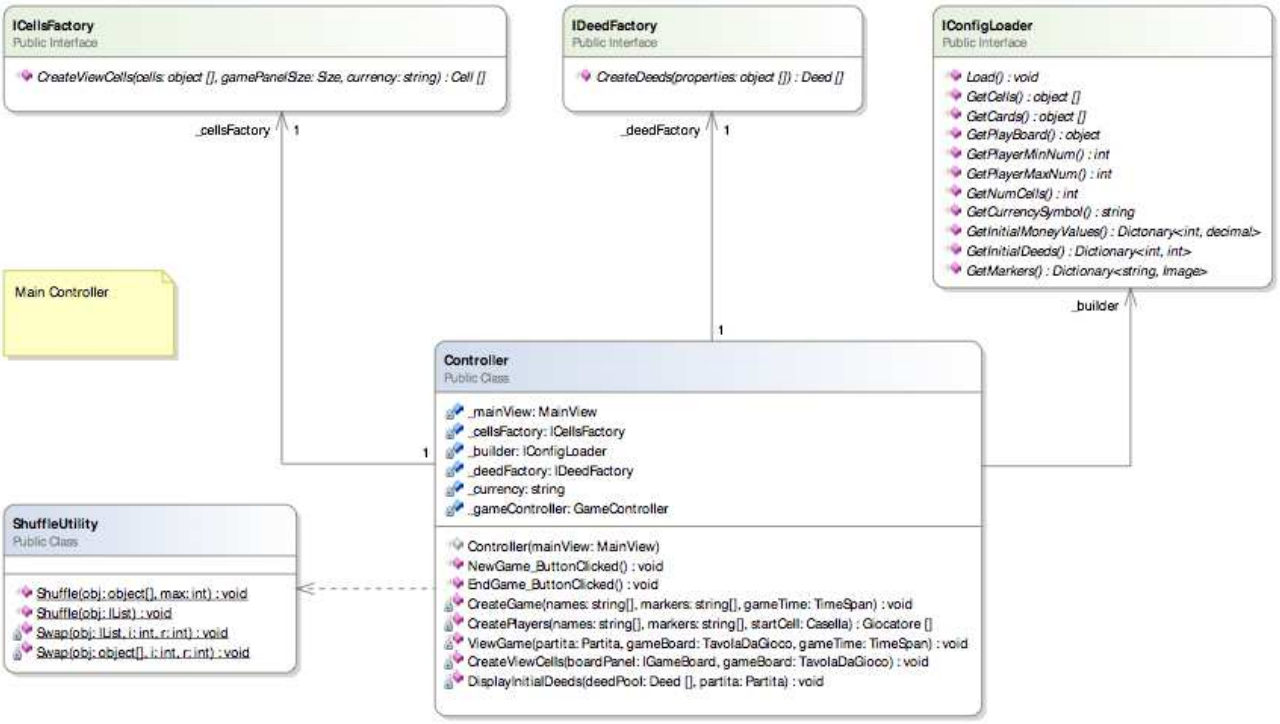
CONTROLLER

Rimanendo nell'ottica del **pattern MVC,** è fondamentale discutere a questo punto anche del **Controller**.
Innanzitutto abbiamo preferito fare uso di un pattern MVC piuttosto che di un pattern MVP per non dover caricare eccessivamente il Presenter di responsabilità.
L'applicazione finale contempla fondamentalmente due stati: uno stato "idle" in cui è visibile la schermata principale e non vi è nessuna partita in corso, e uno stato di gioco, in cui è visibile l'interfaccia di gioco prima descritta.
Nel primo di questi due stati non vi è in realtà necessità di accedere al Model, in quanto questo fa riferimento a una Partita; è tuttavia necessario delegare una qualche entità che non sia la MainView a generare sia il modello, sia l'interfaccia di gioco, quando l'utente desidera creare una nuova partita.

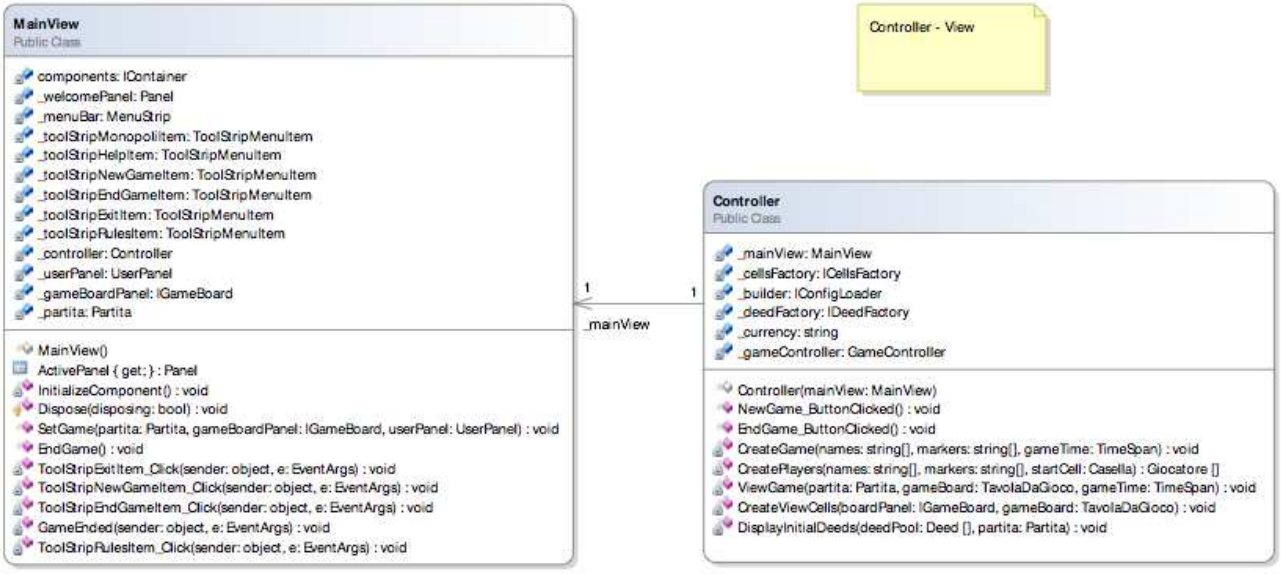
La scelta progettuale è stata quella di suddividere i compiti:

- Creazione di una nuova partita a seguito di input dell'utente
- Gestione di una partita in corso, con tutto ciò che questo comporta

Volendo suddividere i compiti si è pertanto stabilito che conveniva avere due Controller separati, ognuno dei quali svolgesse uno solo dei compiti sopra elencati; si è perciò fatto affidamento a un **Controller** e a un **GameController**.



Il **Controller** (o anche main controller) ha il compito di comunicare esclusivamente con la **MainView**, e deve poter processare gli input dell'utente, in particolare la creazione e la terminazione di una Partita.
Per svolgere appieno il suo compito, deve poter fare affidamento a diverse entità esterne, necessarie per la creazione di tutte le varie parti di una partita.



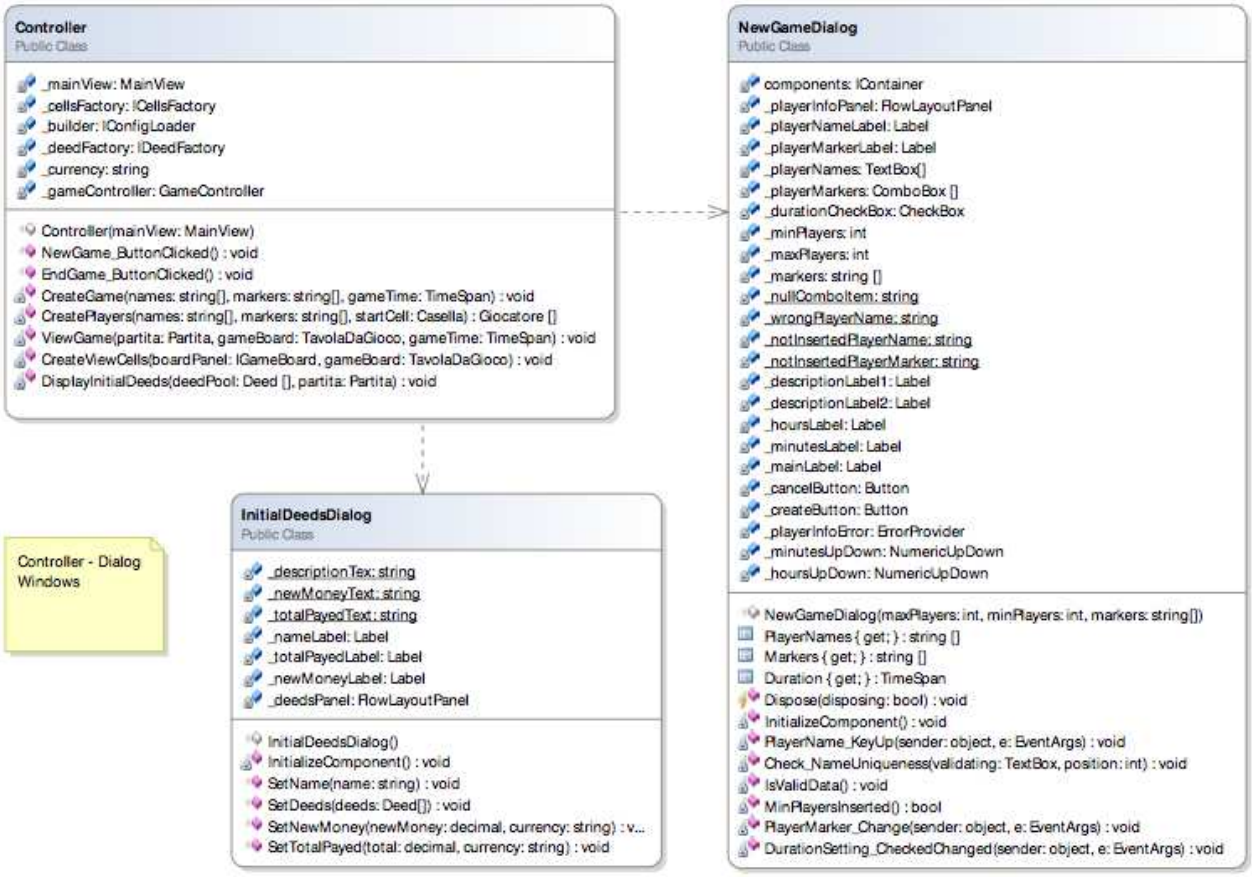
Le classi a cui il Controller può accedere sono:

- **MainView**, che il Controller è in grado di aggiornare e dalla quale vengono invocati i metodi NewGame e EndGame, secondo il pattern MVC
- **IConfigLoader**, il builder già analizzato che ha il compito di caricare in memoria la TavolaDaGioco con tutto il suo contenuto da dati persistenti
- **ICellsFactory**, una factory che permette di inizializzare le caselle grafiche,

date le classi di Model da cui ottenere le informazioni

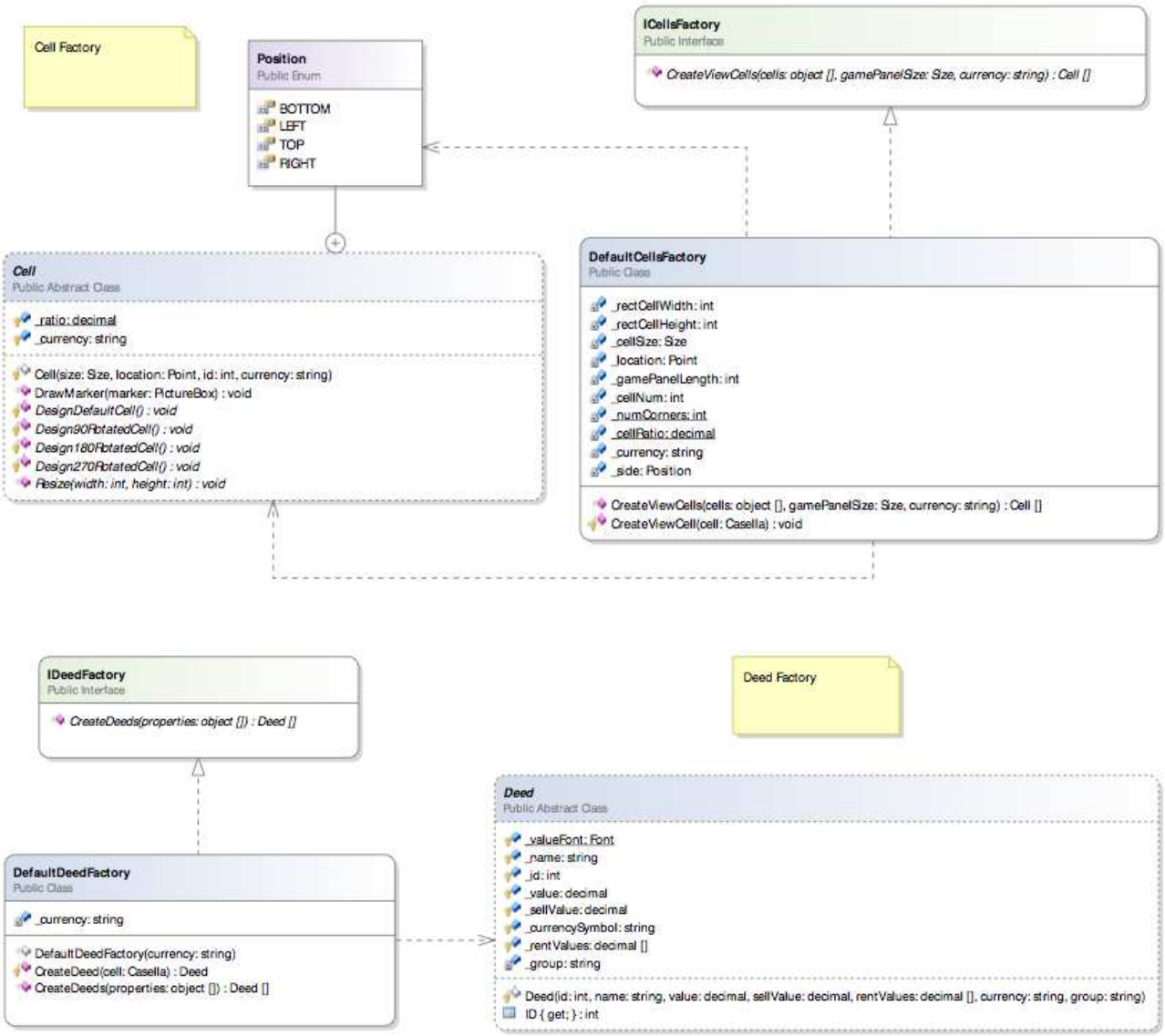
- **IDeedFactory**, una factory che permette di inizializzare i contratti grafici, sempre a partire dalle informazioni contenute nel Model
- La **ShuffleUtility**, necessaria per mescolare i contratti e poterli distribuire casualmente ai giocatori durante la creazione di una partita
- Una finestra di dialogo **NewGameDialog**, che viene aperta quando viene invocato il metodo NewGame_ButtonClicked e permette all'utente: di inserire i nomi dei giocatori che parteciperanno alla partita, di selezionare il segnalino che verrà utilizzato per tutta la partita, di impostare un'eventuale durata della partita
- Una finestra di dialogo **InitialDeedsDialog**, che permette, alla creazione di una partita, di mostrare all'utente i contratti ottenuti casualmente, come da regolamento

Il Controller viene creato dalla MainView all'avvio dell'applicazione. Questo crea all'avvio nuove istanze di **IConfigLoader**, **ICellsFactory**, **IDeedFactory.** Prima ancora di poter processare eventuali input dell'utente, il Controller carica in memoria il modello, grazie al già discusso builder che implementa IConfigLoader; il Model pertanto resterà sempre caricato in memoria, anche quando non vi è una partita in corso. Questa scelta progettuale è dovuta a un requisito non funzionale del cliente, il quale richiede esplicitamente che la creazione di una nuova partita impieghi meno di un secondo. Data questa specifica, si è ritenuto più performante poter utilizzare un modello già pronto alla creazione di una nuova partita, non costringendo il builder a creare tutto quanto da capo ogni volta che viene iniziata una nuova partita.



Nota: la strategia di persistenza adottata porta a dover caricare un XmlDocument e a dover fare il parsing di questo per ottenere tutti i dati necessari alla costruzione del modello; questo processo può impiegare alcune centinaia di millisecondi, come analizzato in fase di testing.

A differenza del modello, tutta l'interfaccia grafica viene costruita solo all'atto della creazione di una partita. Ogni volta che viene creata una partita vengono inizializzate nuovamente sia le celle e i panelli della schermata di gioco, sia i contratti (Deed).



Si è preferito adottare una soluzione del genere, quindi ricreare la grafica a ogni richiesta, per non "saturare" la memoria in casi in cui l'applicazione resti aperta sulla schermata principale ma idle, e l'utente effettui nel frattempo altre operazioni al calcolatore. Trattandosi di elementi grafici non banali (benché bidimensionali), questi tendono a necessitare di una quantità non del tutto indifferente di memoria. Pertanto abbiamo ritenuto più corretto rilasciare le risorse grafiche utilizzate al termine di una partita, e fare quindi più affidamento sulla CPU, non trattandosi di una costruzione eccessivamente complessa.

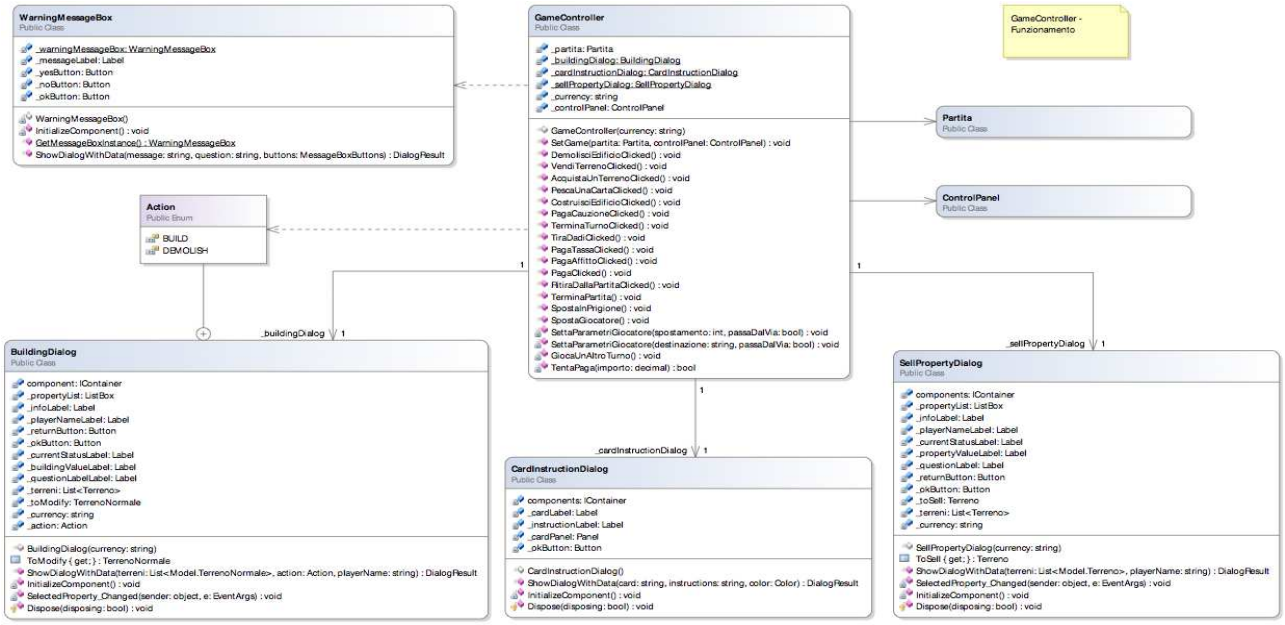
ICellsFactory e **IDeedFactory** vengono rispettivamente implementate dalle factory concrete **DefaultCellsFactory** e **DefaultDeedFactory**. Viene fatto uso del **pattern Factory Method** in entrambi i casi.

In particolare la DefaultCellsFactory è stata progettata specificamente per costruire una tavola da gioco quadrata, quindi la tavola tipica del Monopoli; durante il processo di costruzione, ha il compito di inizializzare le varie **Cell**, passando al loro costruttore i parametri necessari per la creazione: in particolare una struttura Point, una struttura Size e, nella maggior parte dei casi un valore specifico dell'enumerativo Position, innestato nella classe Cell (oltre alle informazioni ottenute dal modello).

Come anche nel caso della costruzione del modello, l'istanziazione di tutte le classi in gerarchia della View (quindi Deed e Cell) avviene tramite Reflection. Si ottengono dal modello direttamente gli argomenti necessari da passare al costruttore come array di object, invocando i metodi relativi GetContratto e GetParametriCella.
Con questa soluzione si riescono a evitare costrutti come switch e batterie di if/else.

L'intera gestione della partita è invece affidata al **GameController**. Alla creazione della partita viene chiamato dal Controller il metodo SetGame sul GameController, al quale vengono passati il **Model** e il **ControlPanel,** ovvero la View con il quale l'utente può interagire durante la sessione di gioco.

Date le specifiche Model-View-Controller, il **GameController** da quel momento in avanti ha la mansione di gestire le chiamate provenienti dal ControlPanel, a seguito di richieste dell'utente; fungendo da Dispatcher è dunque in grado di invocare i metodi giusti sul Model per modificarne lo stato.



Sebbene tutte le azioni scatenate dall'utente vengano raccolte dal ControlPanel, non tutte le finestre di dialogo descritte nei casi d'uso possono essere direttamente aperte e gestite da esso.
Le finestre di dialogo **BuildingDialog**, **CardInstructionDialog** e **SellPropertyDialog**, tutte quante classi derivate da Form, possono essere aperte solamente in seguito a check e calcoli effettuati dal GameController; perciò è proprio questa classe a mantenere in memoria queste tre classi grafiche. Sono istanze statiche, il cui tempo di vita è pari a quello della partita corrente:

- il **BuildingDialog** permette la costruzione/distruzione di edifici su delle caselle (secondo le specifiche di progetto solamente sui Terreni), in base al valore dell'enumerativo **Action** passato nel metodo ShowDialogWithData; il GameController a ogni richiesta cerca solamente le caselle effettivamente edificabili, non permettendo all'utente di selezionare caselle non edificabili. Qualora l'utente decida di portare a termine l'operazione, il GameController deve recuperare la selezione dell'utente stesso e chiamare il metodo relativo nel Model.
- Il **CardInstructionDialog** mostra all'utente le istruzioni di una carta pescata, dopo che questa è stata opportunamente reperita dal modello.
- Il **SellPropertyDialog**, infine, consente la vendita di una casella di cui si è in possesso. La conferma dell'utente fa sì che il GameController recuperi la selezione dell'utente e operi conseguentemente sul modello.

Per semplicità, è stato reso possibile all'utente finale utilizzare in qualsiasi istante, durante lo svolgimento di una partita, i pulsanti collegati al ControlPanel che aprissero una delle sopra elencate finestre di dialogo statiche (sia quelle presenti in ControlPanel, sia quelle presenti nel GameController): qualora sia impossibile effettuare una delle operazioni richieste, come ad esempio vendere una proprietà

quando non se ne possiedono, il sistema apre ugualmente la finestra di dialogo relativa, disabilitando tuttavia il pulsante di conferma interno ad essa. Le finestre di dialogo che devono popolare una ListBox con dei contenuti ottenuti dal modello, accettano delle liste vuote, dunque l'utente non potrà selezionare nulla in caso sia stata passata una lista senza contenuti.

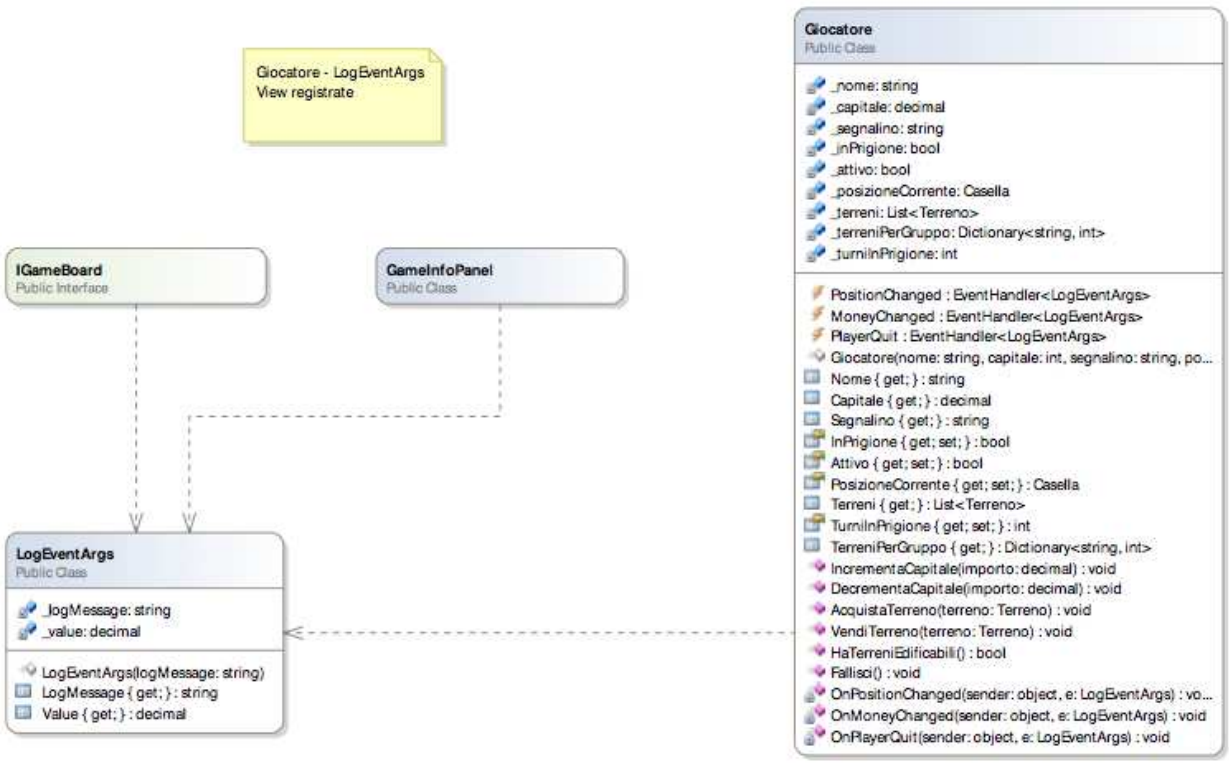
Al contrario, come già esplicitato nelle note delle descrizioni dei casi d'uso, tutte le altre funzioni di gioco vengono atuomaticamente disabilitate se non è possibile farne uso: se non è possibile selezionare il pulsante Tira Dadi per esempio, questo verrà disabilita dal sistema, tornando attivo soltanto quando saranno soddisfatte tutte le precondizioni.

Come indicato al cliente, è stata implementata anche una finestra di dialogo **WarningMessageBox;** questa finestra di dialogo compare quando è necessario chiedere conferma all'utente di una determinata operazione, come l'acquisto di un terreno o, nella maggiore parte dei casi, il fallimento. Si è voluto evitare di far fallire un giocatore automaticamente qualora esso non riuscisse a pagare una determinata somma durante il suo turno; al giocatore in questione viene invece data la facoltà di scegliere se fallire oppure no. Nel caso il giocatore opti per la seconda scelta, egli avrà la possibilità di vendere terreni e/o edifici per arrivare eventualmente a una somma pari a quella da pagare.
La **WarningMessageBox** è un **Singleton**, e viene usata come alternativa al **MessageBox** di .NET tradizionale. Questo perché deve essere possibile chiudere arbitrariamente l'istanza di WarningMessageBox dall'esterno (cosa non possibile con MessageBox tradizionale); chi si occupa di quest'operazione è la MainView, che deve poter chiudere le WarningMessageBox al termine di una partita.

MODEL-VIEW-CONTROLLER
Trattiamo ora delle relazioni tra Model, View e Controller più nel dettaglio.
Come già detto, il **Controller** principale ha il compito di costruire modello e interfaccia di gioco quando l'utente inizia una nuova partita. Il **GameController** controlla unicamente lo svolgimento della partita e conosce solamente il ControlPanel come entità di View, mentre il Controller conosce esclusivamente la MainView, pur avendo creato anche tutti gli altri pannelli.

Le entità di View che si registrano agli eventi del Model sono la **MainView**, il **ControlPanel** e il **GameInfoPanel**.
Le classi di Model che sono in grado di notificare le View in caso di cambiamenti di stato sono:

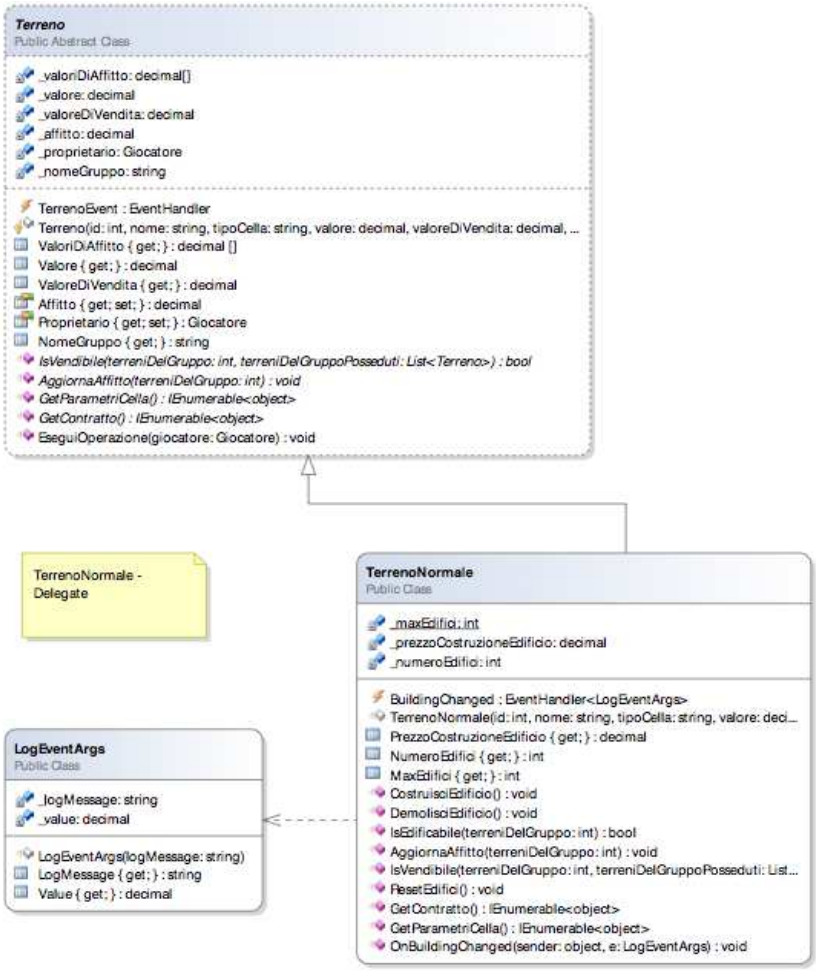
- **Partita**, in grado di lanciare gli eventi PlayerChanged, GameEnded e GameTimer, a cui si registrano la MainView e il GameInfoPanel.



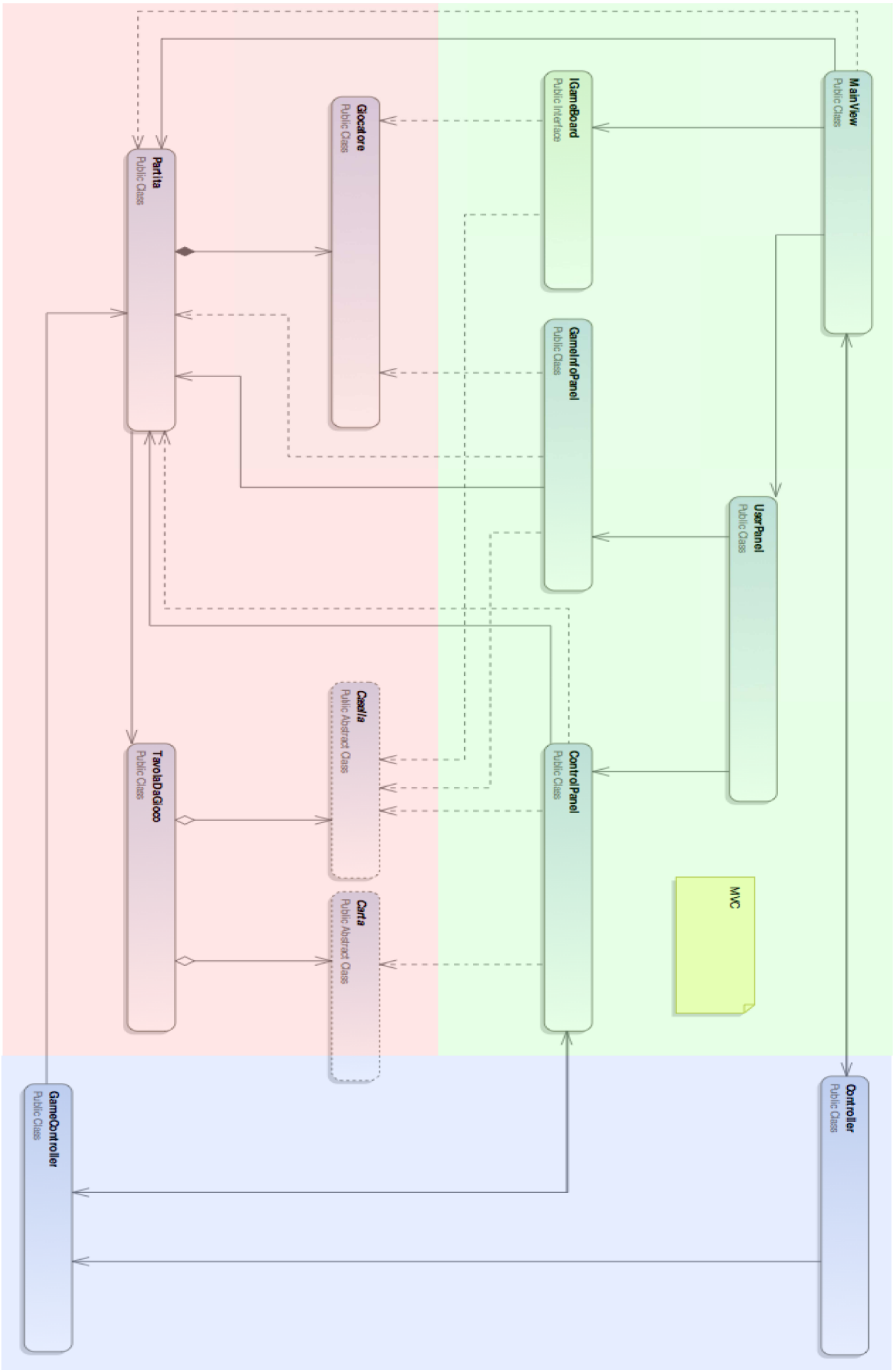
- **Giocatore**, in grado di lanciare gli eventi MoneyChanged, PositionChanged, PlayerQuit; il GameInfoPanel è registrato a tutti questi eventi di Giocatore, per poter fornire dettagli su ogni cambiamento di stato di un Giocatore, mentre la GameBoard si registra esclusivamente a PositionChanged e PlayerQuit.
- **Casella,** la cui responsabilità è notificare la View ogni qualvolta un Giocatore si ferma con il proprio segnalino su di essa. Il **Template Method pattern** utilizzato per la gerarchia di Casella serve appunto per questo scopo. Quando un giocatore modifica la propria posizione attuale, esegue sempre il metodo EseguiOperazione, presente in tutte le caselle. Questo metodo provvederà ad effettuare le operazioni necessarie e a notificare tutti gli osservatori, nel caso di questo prototipo la View; così facendo, la View abiliterà i pulsanti relativi alla casella che ha lanciato l'evento. Non tutte le caselle comportano azioni specifiche, quindi si è preferito definire un evento per ogni casella concreta, piuttosto che definirne uno nella classe astratta da cui tutte le caselle ereditano.
- Le sottoclassi di **Carta**, per le quali vale lo stesso discorso delle caselle: non tutte quante comportano azioni specifiche, dunque le classi concrete di carta che devono essere in grado di notificare il ControlPanel hanno dichiarato al loro interno un evento pubblico specifico.
- **TerrenoNormale**, la cui superclasse Terreno è in grado di lanciare l'evento TerrenoEvent, con il comportamento appena descritto; TerrenoNormale può tuttavia notificare la GameBoard anche in caso vengano costruiti edifici su di esso tramite l'evento BuildingChanged.

Si è scelto di implementare una sottoclasse di EventArgs, definita **LogEventArgs**, che viene utilizzata da alcuni degli EventHandler appena citati.
Lo scopo di questa implementazione è fornire al GameInfoPanel informazioni aggiuntive del modello, in modo da poterle stampare sulla LogBox all'interno dello stesso pannello, che, come già precedentemente esplicitato, ha come unica funzione quella di collegarsi agli eventi del Model e aggiornarsi in seguito a una notifica.

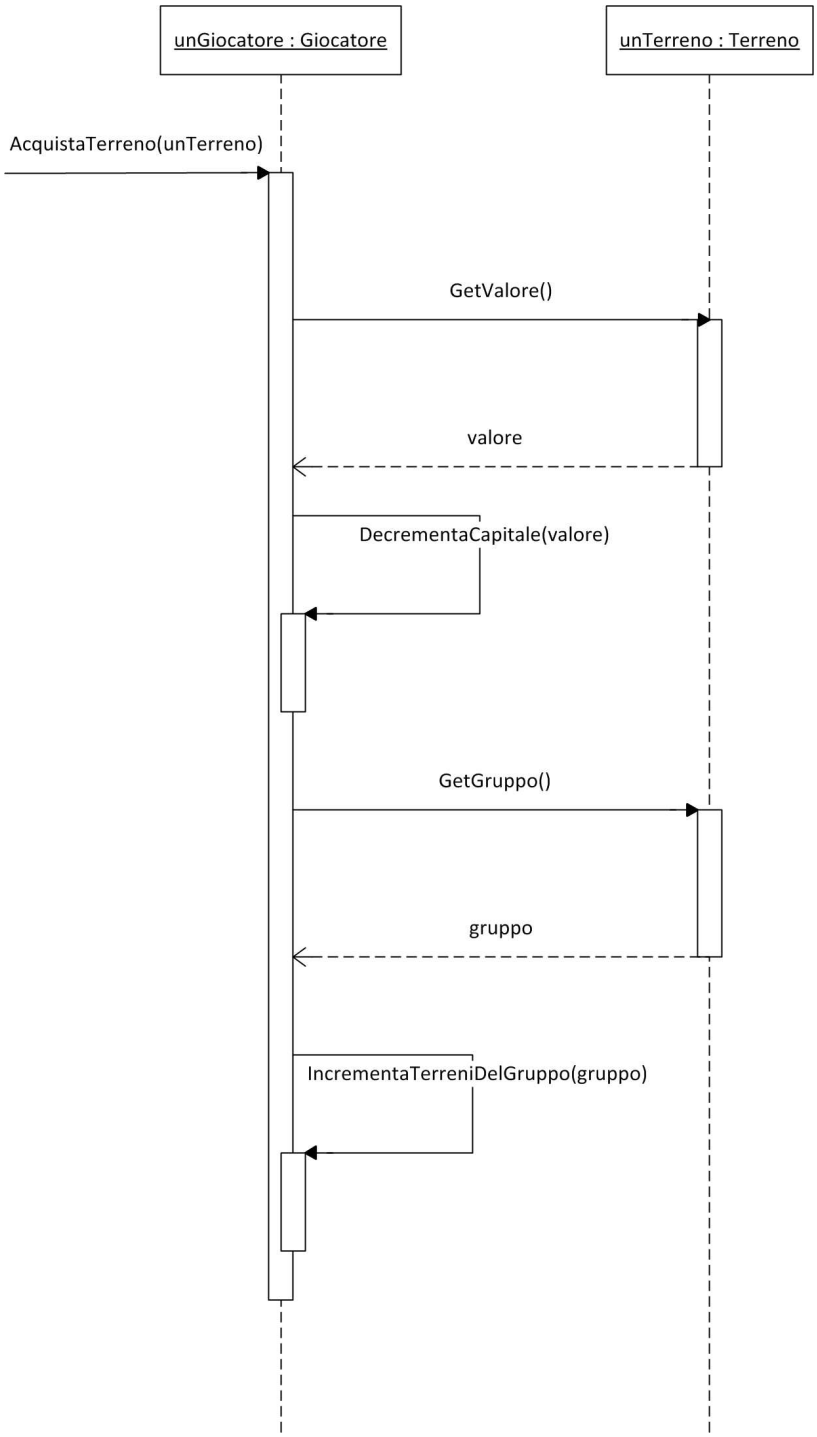
Nella pagina successiva è riportato lo schema completo della relazione MVC tra le varie classi.



Considerazioni sulla progettazione:
Durante l'intera fase di progettazione si è sempre tenuto conto delle specifiche funzionali e non funzionali del Cliente, dei casi d'uso e di tutti gli scenari analizzati in precedenza.
Questo ci ha permesso di non distanziarci troppo dall'analisi, e soprattutto ci ha permesso di effettuare immediatamente modifiche nella parte di analisi quando venivano effettuate scelte di progetto con ripercussioni su questa.
Come già spiegato, abbiamo mirato a rendere l'intero software poco fragile, non particolarmente rigido, e facilmente estendibile; in particolare si è presa in considerazione la possibilità che il cliente commissionasse in futuro differenti versioni del Monopoli, con regole e tavole da gioco differenti. Si è fatto uso dei principali principi di progettazione per raggiungere questo risultato.


MainView
Public Class
IGameBoard
Public Interface
Giocatore
Public Class
Partita
Public Class
GameInfoPanel
Public Class
UserPanel
Public Class
ControlPanel
Public Class
Casella
Public Abstract Class
Carta
Public Abstract Class
TavolaDaGioco
Public Class
MVC
Controller
Public Class
GameController
Public Class

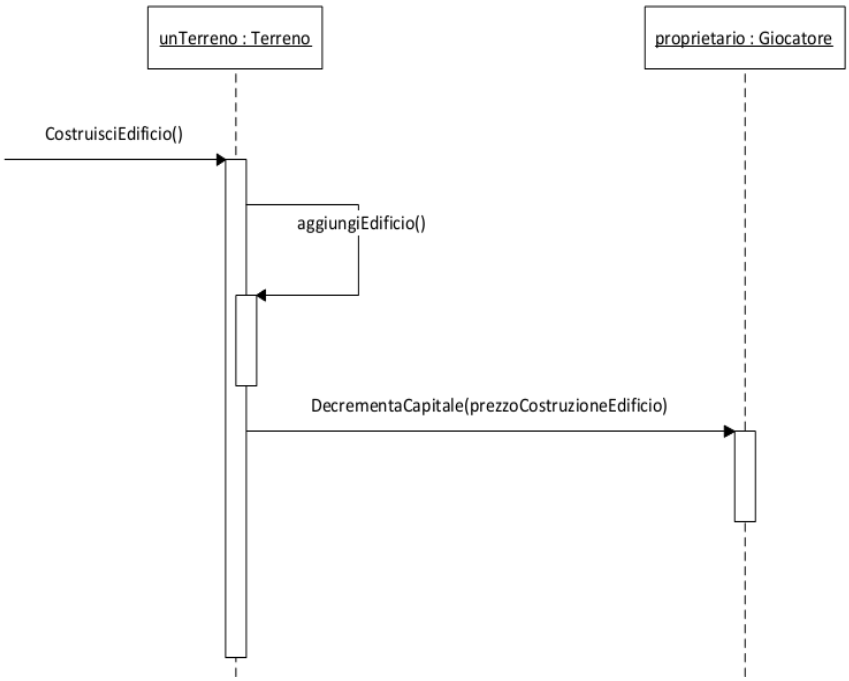
ACQUISTA TERRENO
(MODIFICATO IN BASE A SCELTE PROGETTUALI)



Come scelta progettuale, le informazioni contenute nel Contratto, sono accorpate nel Terreno; pertanto lo scambio di informazioni avviene fra Giocatore e Terreno.
Le operazioni si svolgono nel seguente ordine:

- Giocatore ottiene il valore del terreno che desidera acquistare;
- Giocatore decrementa il proprio capitale del valore ottenuto;
- Giocatore ottiene il gruppo del terreno che sta acquistando;
- Giocatore incrementa di una unità il contatore di terreni posseduti del gruppo del terreno da acquistare.

COSTRUISCI EDIFICIO
(MODIFICATO IN BASE A SCELTE PROGETTUALI)



L'operazione di costruzione di un edificio è spostata in Terreno.
L'edificabilità del terreno non è verificata da CostruisciEdificio(), in quanto il GameController mostra al giocatore una vista che gli consente di selezionare uno fra i soli terreni su cui esso può edificare.
Terreno ha comunque al suo interno un metodo isEdificabile(), chiamato dal GameController per realizzare la vista sopra citata.

CONSIDERAZIONI FINALI

Considerazioni sul Testing

Il Videogioco è stato ampiamente testato durante tutta la fase di sviluppo, con svariate ore di alpha testing finale.
In quanto prototipo si è provveduto ad eliminare i bug principali dovuti a errori di progettazioni degli algoritmi e a errata coordinazione del GameController.
Per testare il grado di portabilità, è stato testato su differenti macchine.
Trattandosi di un videogioco in 2D non si è tenuto conto delle differenti GPU ma soltanto delle CPU.
- La risoluzione video minima per poter giocare è di 1024x768 pixel.

MACCHINA	RISOLUZIONE VIDEO	PERFORMANCE
Intel Core i5 2.53 Ghz 8 GB Ram Windows 7	1680x1050	OTTIMA
Intel Core 2 Duo 2.2 Ghz 4 GB Ram Windows 7	1366x768	OTTIMA
Intel Core 2 Duo 2.4 Ghz 2 GB Ram Windows Vista	1440x900	OTTIMA

Le performance complessive sono ottime, indipendentemente dal tipo di CPU utilizzata; è tuttavia consigliabile una risoluzione video più alta di quella minima.
La quantità di RAM media utilizzata dall'applicazione varia dai 15Mb ai 35Mb.

Non è stato effettuato testing vero e proprio su una macchina con sistema operativo Windows XP in fase di alpha testing.
Non avendo ancora implementato un Windows Installer per l'installazione del software (in quanto prototipo), non viene automaticamente installato il Framework .NET 3.5 su macchine Client. Se questo non è presente, all'avvio dell'eseguibile Monopoli.exe l'applicazione non viene lanciata, restituendo un messaggio di errore.

Strumenti utilizzati per il compimento del progetto:

- **Microsoft Visual Studio Professional 2010** – utilizzato per lo sviluppo del codice del prototipo, comprese interfacce grafiche
- **Microsoft Visio 2010** – strumento di design utilizzato per tutti i diagrammi e le classi di analisi
- **NClass** – utilizzato per il design delle classi di progettazione C#
- **OpenOffice** – utilizzato per la stesura dei requisiti e la documentazione
- **Adobe Photoshop CS5** – strumento utilizzato per la gestione delle immagini