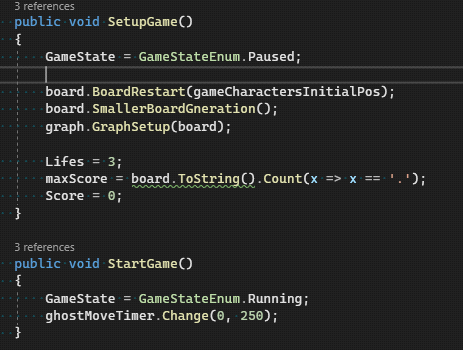
PAC-Man

Prezentare proiect

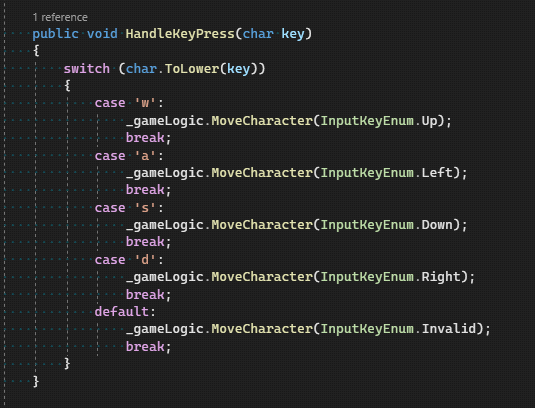
La începerea jocului primul lucru care trebuie realizat este harta, aceasta este una standard. Pe baza acesteia se realizează un graf, acesta este folosit pentru calcularea mișcărilor fantomelor către Pac-Man. Acestea fac inițial o mișcare la fiecare sfert de secundă, dar acest lucru se poate schimba pe viitor.



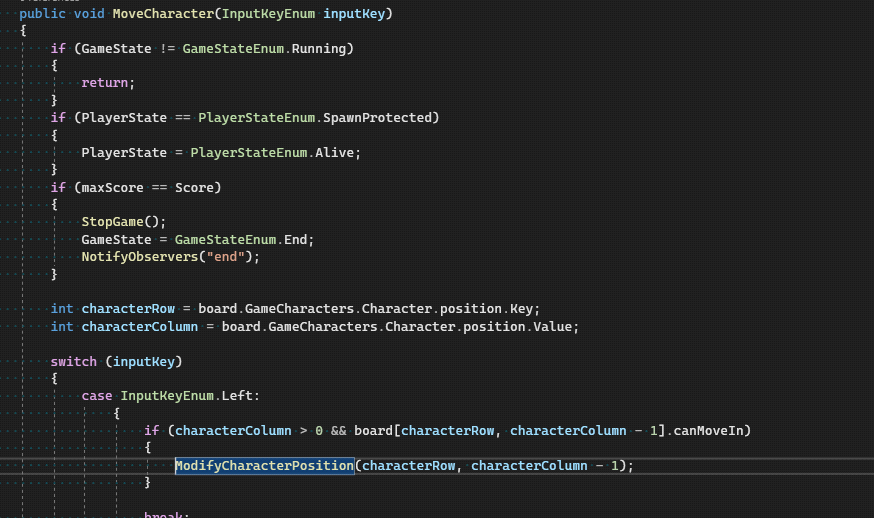
Aceste metode sunt apelate din UI prin instanța clasei IGameLogic. Remarcăm că înainte de metoda StartGame, sunt apelate 2 metode care au rolul de a construii Board-ul în partea de UI. Timerul este folosit pentru a verifica daca sunt schimbări în partea de UI(s-a mutat caracterul/ fantomele).



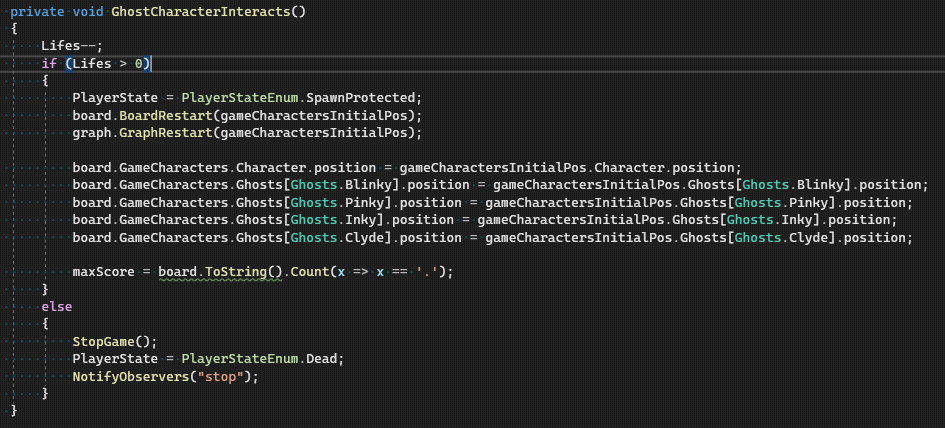
La fiecare miscare realizata de jucator este apelata MoveCharacter din clasa GameLogic.



In aceasta metoda sunt efectuate mai multe verificari, iar daca mutarea este permisa aceasta este realizata.



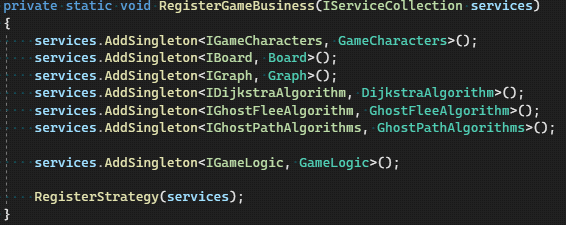
Pe parcursul jocului sau in metoda incadrata de mai sus (ModifyCharacterPosition) poate fi apelata metoda de mai jos care marcheaza faptul ca fantoma a intercationat cu caracterul. În acest caz, fie jocul este pierdut daca caracterul a ramas fara vieti, fie se reia jocul cu caracterul si fantomele repuse in pozitia initiala, dar cu scorul neschimbat.



Design pattern-uri:

* Singleton

Este folosit pentru a reține instanțele de pe partea de back-end folosite în front-end.



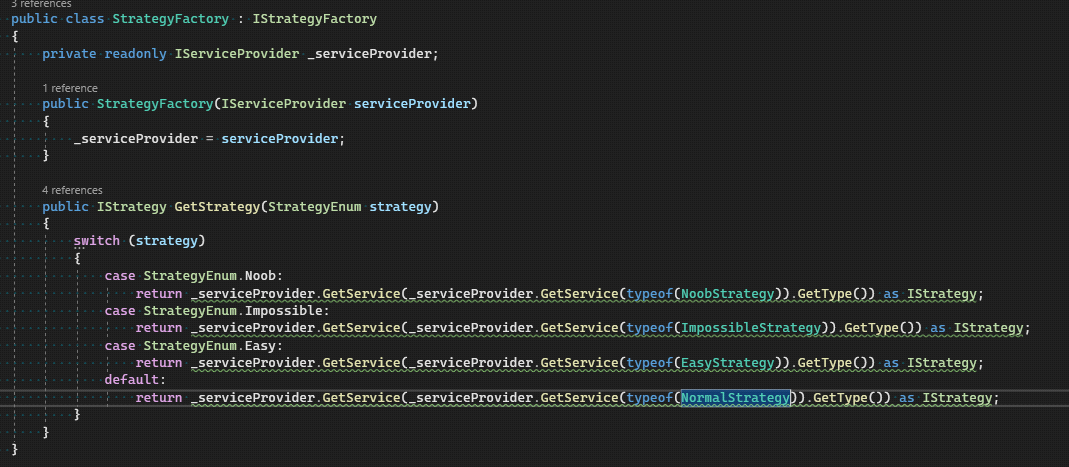
* Observer

Este folosit pentru înștința toate componentele proiectului despre starea jocului(Lobby, Running, Pause, Stop, etc.)



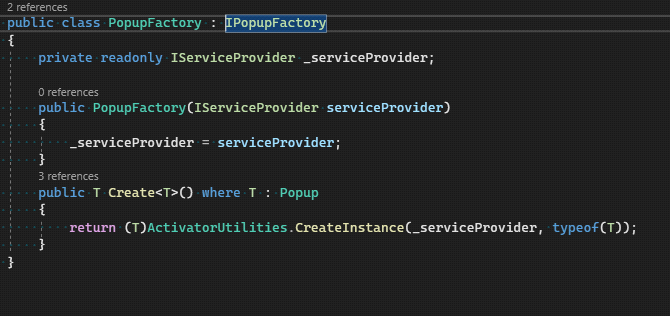
* Strategy

Este folosit pentru schimba modul de dificultate al jocului, acest lucru este realizat prin realizarea mutarilor fantomelor, iar in cazul lui NoobStrategy fantomele chiar fug de jucator.



* Factory

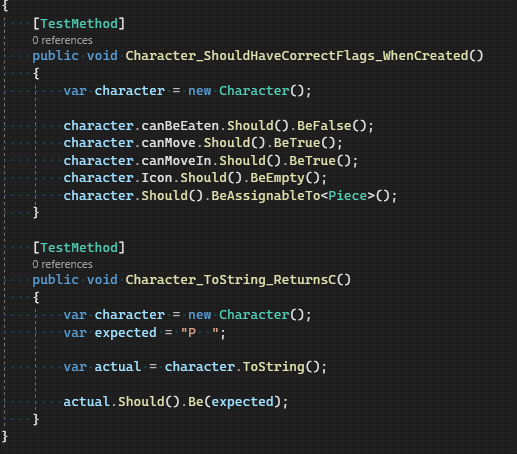
Este folosit pentru a crea instanțe a diferitor tipuri de obiecte, in cazul de mai jos, al ferestrelor de tip Popup cum ar fi cele de pause, respectiv, afisarea controalelor.



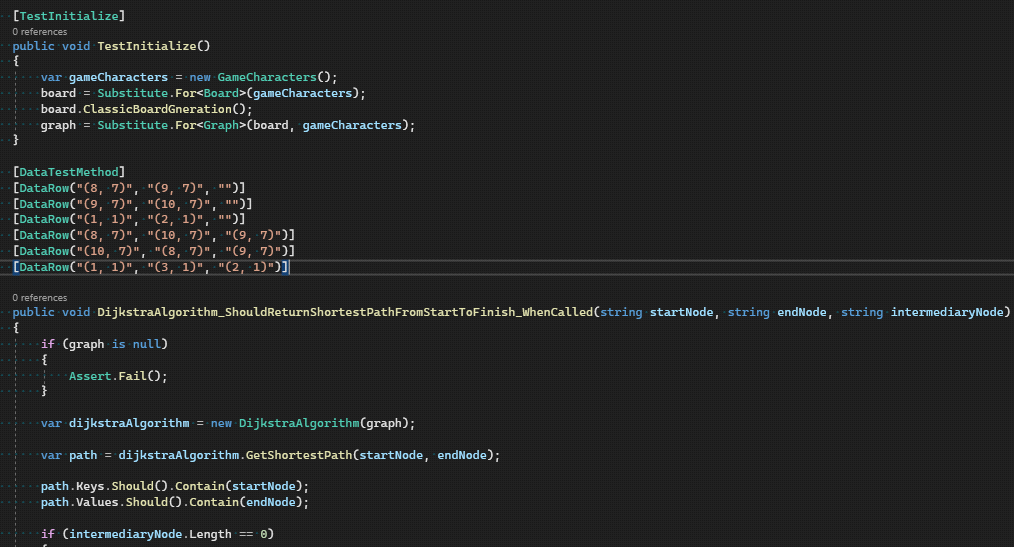
UnitTeste

Am realizat teste in general pe partea de back-end pentru algoritmii de îndrumare a fantomelor, pe partea de Strategy și pe modelele folosite .

Am realizat atât teste simple



cât și de tip mocking



Diagrame:

* UseCase

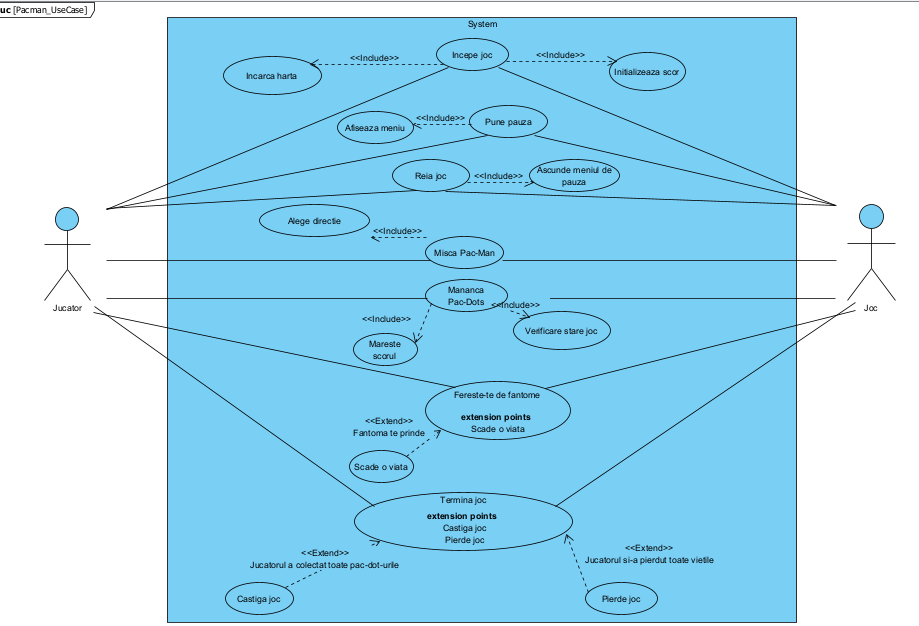


Diagrama UseCase prezintă use case-urile jocului, actorii (jucătorul și jocul), și relațiile dintre use case-uri (prin extend sau include), precum și cele dintre use case-uri și actori. Din această diagramă se poate observa faptul că jucătorul alege să realizeze o serie de acțiuni, iar în funcție de alegerile acestuia jocul reacționează.

* Class :
* Business

Această parte de Business a aplicației este compusă din mai multe componente:

- modelele de obiecte de bază (Piece și clasele sale derivate)

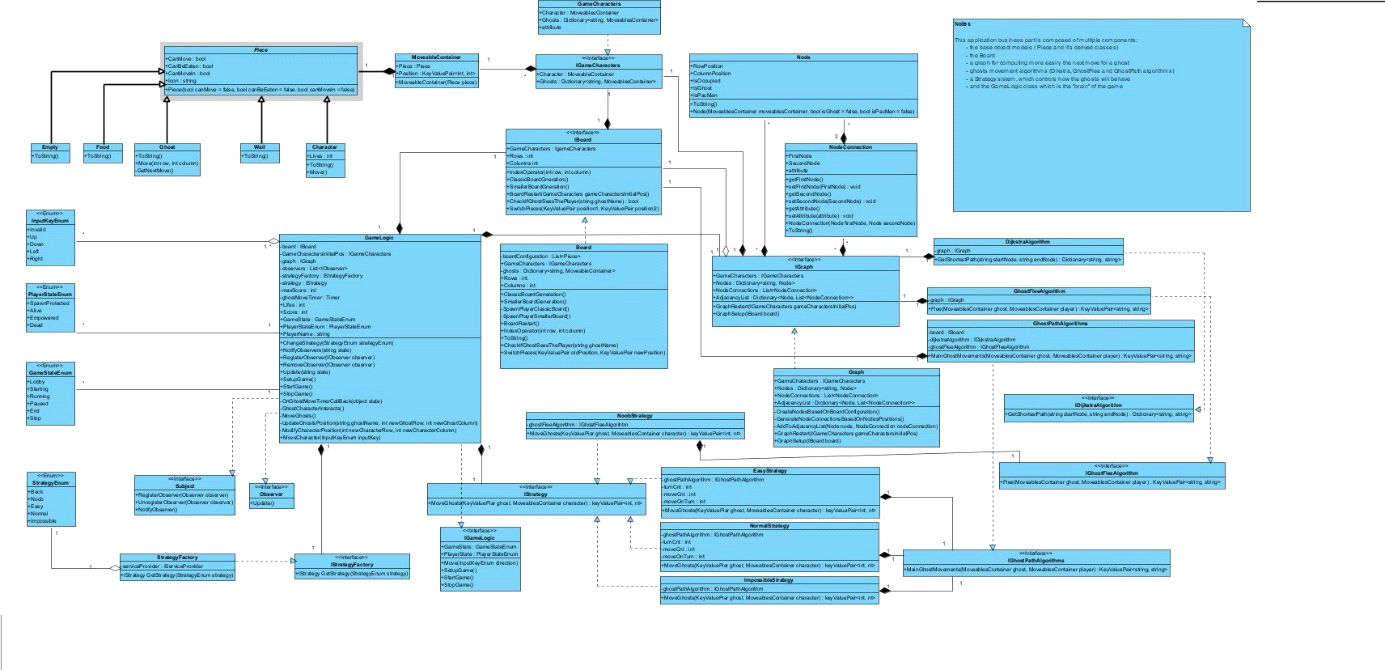
- Board-ul

- un grafic pentru a calcula mai ușor următoarea mișcare pentru o fantomă

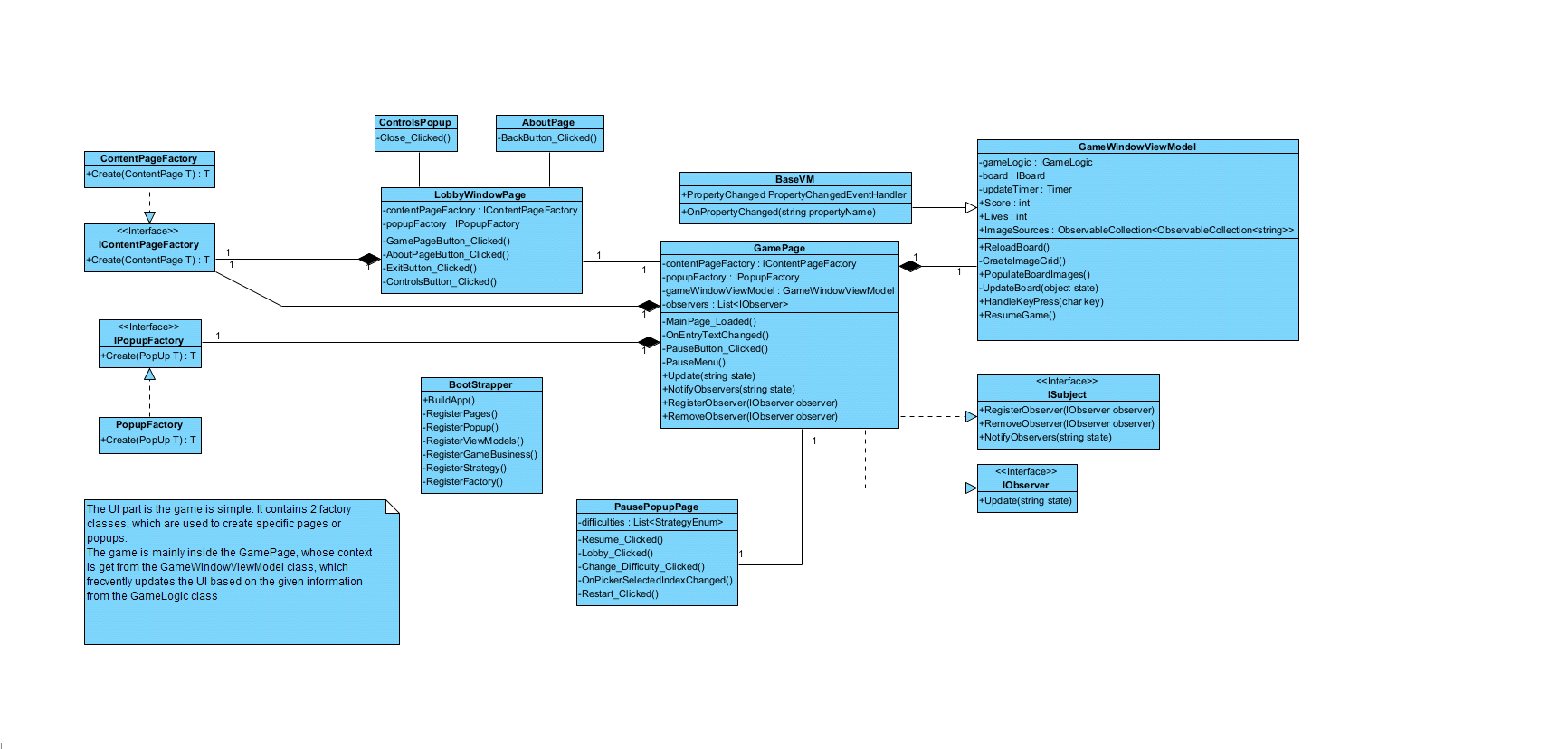
- algoritmi de mișcare a fantomelor (algoritmi Dikstra, GhostFlee și GhostPath)

- un sistem de strategie, care controlează modul în care se vor comporta fantomele

- și clasa GameLogic care este „creierul” jocului

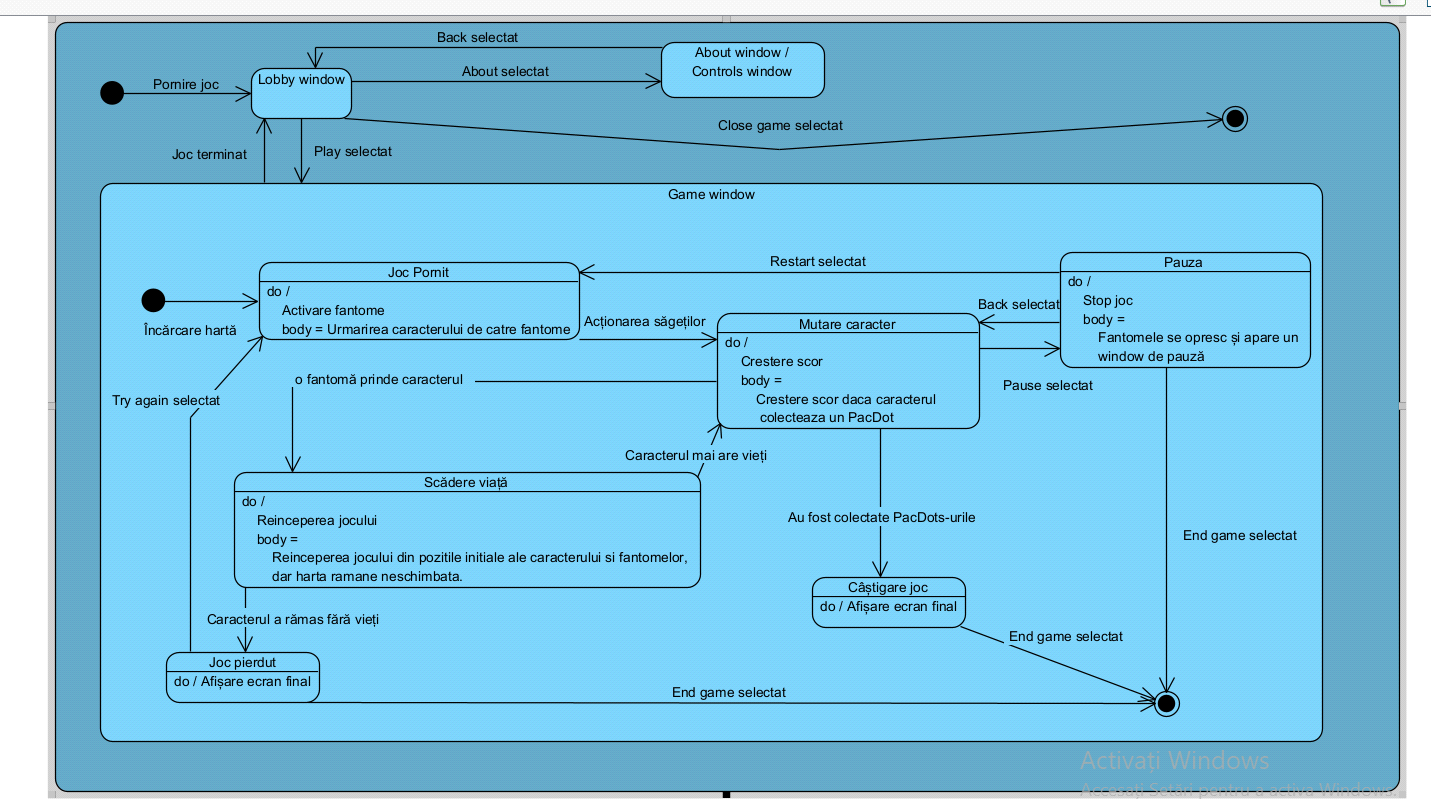


* UI

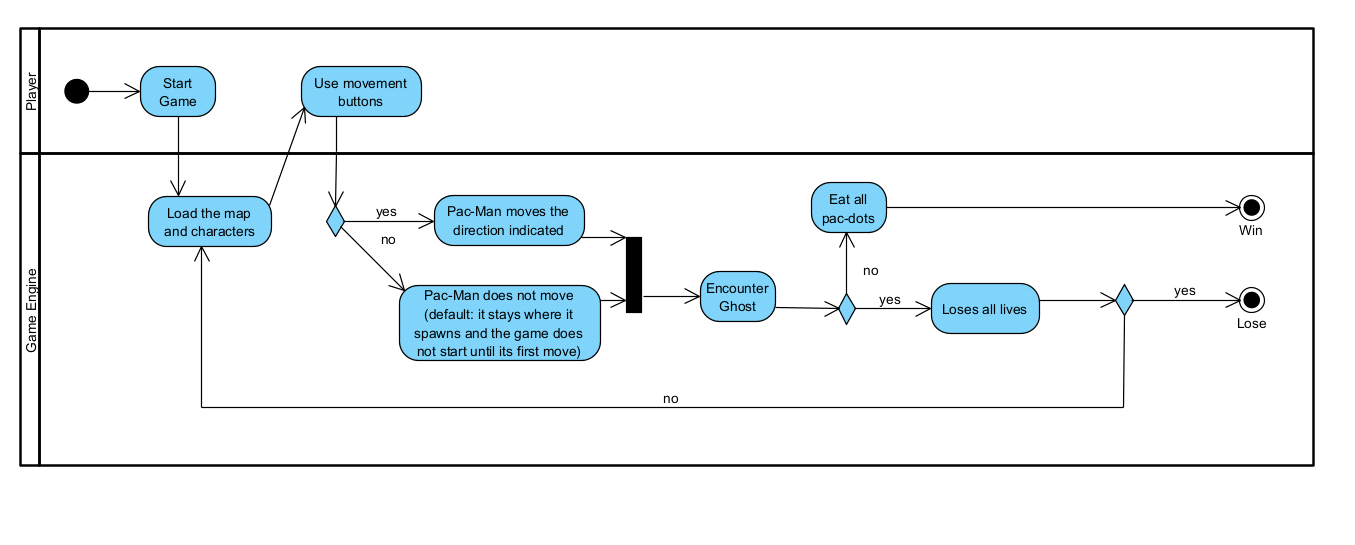
Partea UI conține 2 clase factory, care sunt folosite pentru a crea anumite pagini sau ferestre pop-up. Jocul se află în principal în GamePage, al cărui context este preluat din clasa GameWindowViewModel, care actualizează frecvent interfața de utilizare pe baza informațiilor date din clasa GameLogic.

* State Chart

În diagrama aceasta observăm cursul jocului, care sunt etapele prin care trecem și ce duce la acele etape. În principal jocul începe la acționarea butonului play, iar apoi jucătorul încearcă să colecteze toate Pac-Dots-urile și fie reușește și căștigă, sau își pierde viețile încercănd.



* Activity

 Diagrama de activitate prezintă ordinea firească a derulării jocului, cum funcționează acesta, precum și modurile în care un jucător poate câștiga sau pierde jocul, prin intermediul acțiunilor sau punctelor de decizie. De altfel, se pot observa și actorii care realizează fiecare acțiune, prin intermediul culoarelor (swimlane-urilor). Actorii prezenti în această diagramă sunt jucătorul (player) și game engine-ul.

* Sequence

Fluxul aplicației este simplu:

- deschide jocul

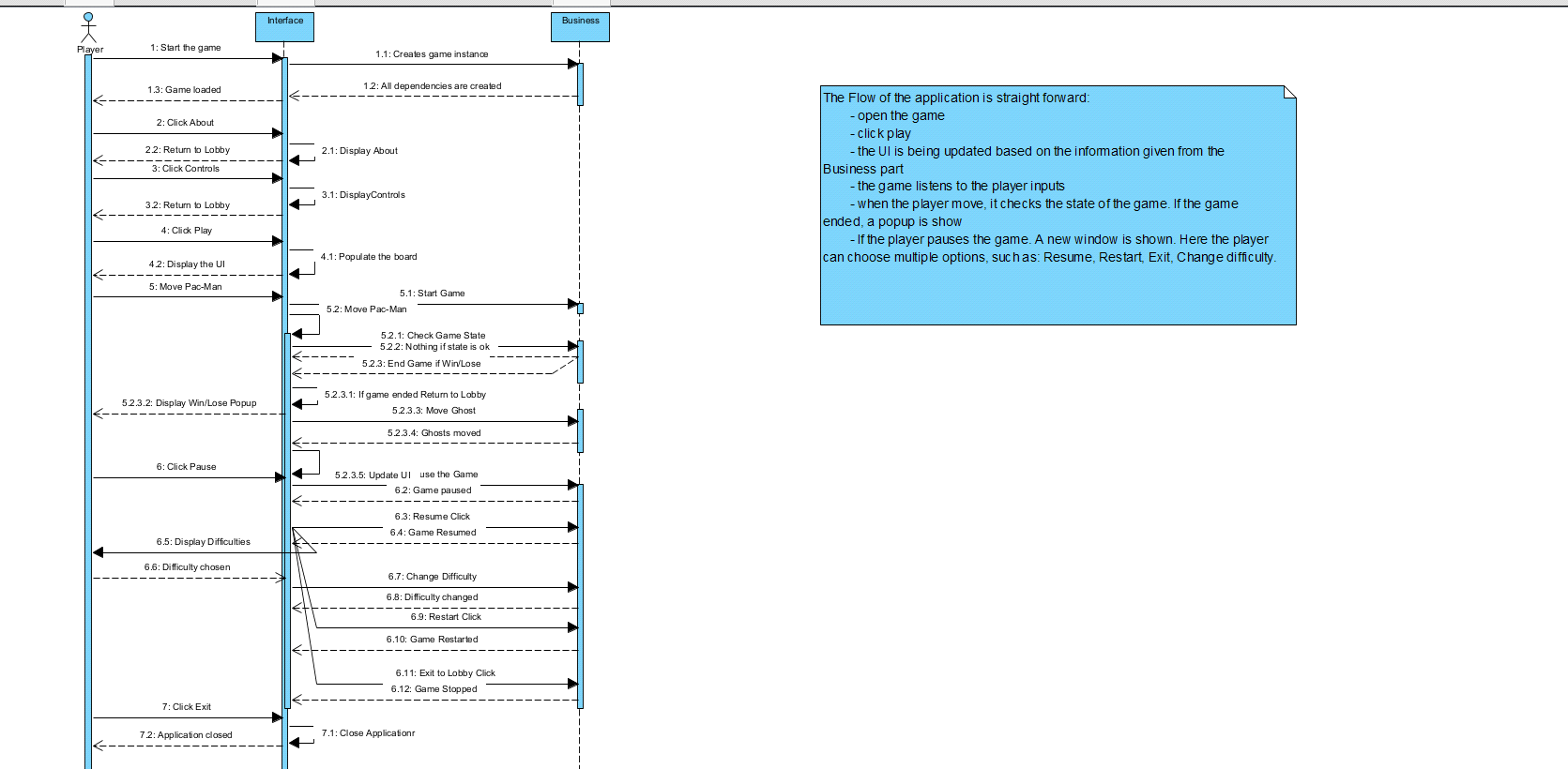
- faceți clic pe play

- interfața de utilizare este actualizată pe baza informațiilor furnizate din partea Business

- jocul ascultă intrările jucătorului

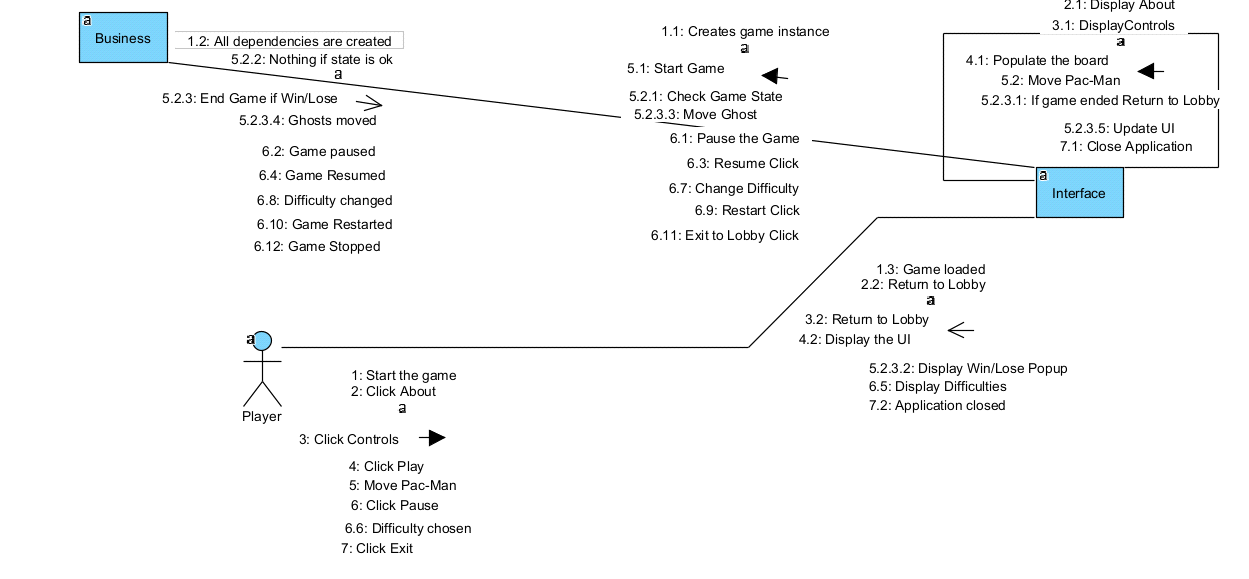
- când jucătorul se mișcă, verifică starea jocului. Dacă jocul s-a încheiat, se afișează o fereastră pop-up

- Dacă jucătorul întrerupe jocul. Este afișată o nouă fereastră. Aici jucătorul poate alege mai multe opțiuni, cum ar fi: Reluare, Repornire, Ieșire, Modificare dificultate.



* Collaboration

În aceasta diagramă observăm cum se realizează comunicare dintre cele mai 3 componente, și anume partea de business, interfața și jucătorul. Jucătorul declanșează în general toate acțiunile, iar între UI și business se desfășoară mai multe procese până la efectul așteptat de jucător.



Membrii:

* Ungureanu Alexandru
* Vulcănean Ionel
* Stroescu Cristian