

2. Prezentarea aplicației dezvoltate

2.2.2 Interfața grafică

Punctul de start al aplicației trebuie să fie ecranul de autentificare(figura 10) din care utilizatorul poate alege să se înregistreze, să se autentifice sau să intre ca și vizitator.

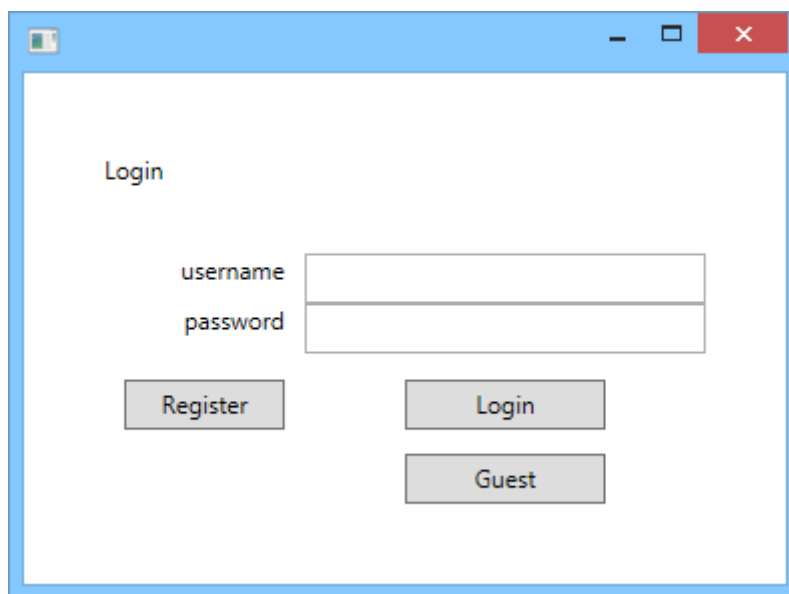


Figura 10

Ecranul de autentificare

Următorul pas după autentificare este afișarea ecranului de joc, acesta constă din 4 secțiuni: meniul cu opțiunile utilizatorului, tabla de șah, zona pentru notificări și un istoric cu lista mutărilor efectuate(figura 11). Meniul trebuie să includă opțiunile pentru salvarea jocului curent, încărcarea unui joc salvat și pentru anularea ultimei mutări efectuate. Utilizatorul va interacționa cu tabla de șah hotărând ce piesă vrea să mute și poziția pe care va ajunge această piesă, în acest moment istoricul de mutări se va actualiza și va afișa poziția de start, poziția unde a ajuns și codul piesei care a fost mutată. Zona de notificări este folosită pentru a anunța culoarea jucătorului ce trebuie să mute și pentru diferite evenimente ce au loc în timpul meciului, de exemplu faptul că o mutare nu este permisă din cauza poziției de șah sau faptul că jocul s-a terminat,

2. Prezentarea aplicației dezvoltate

moment în care trebuie anunțat câștigătorul.

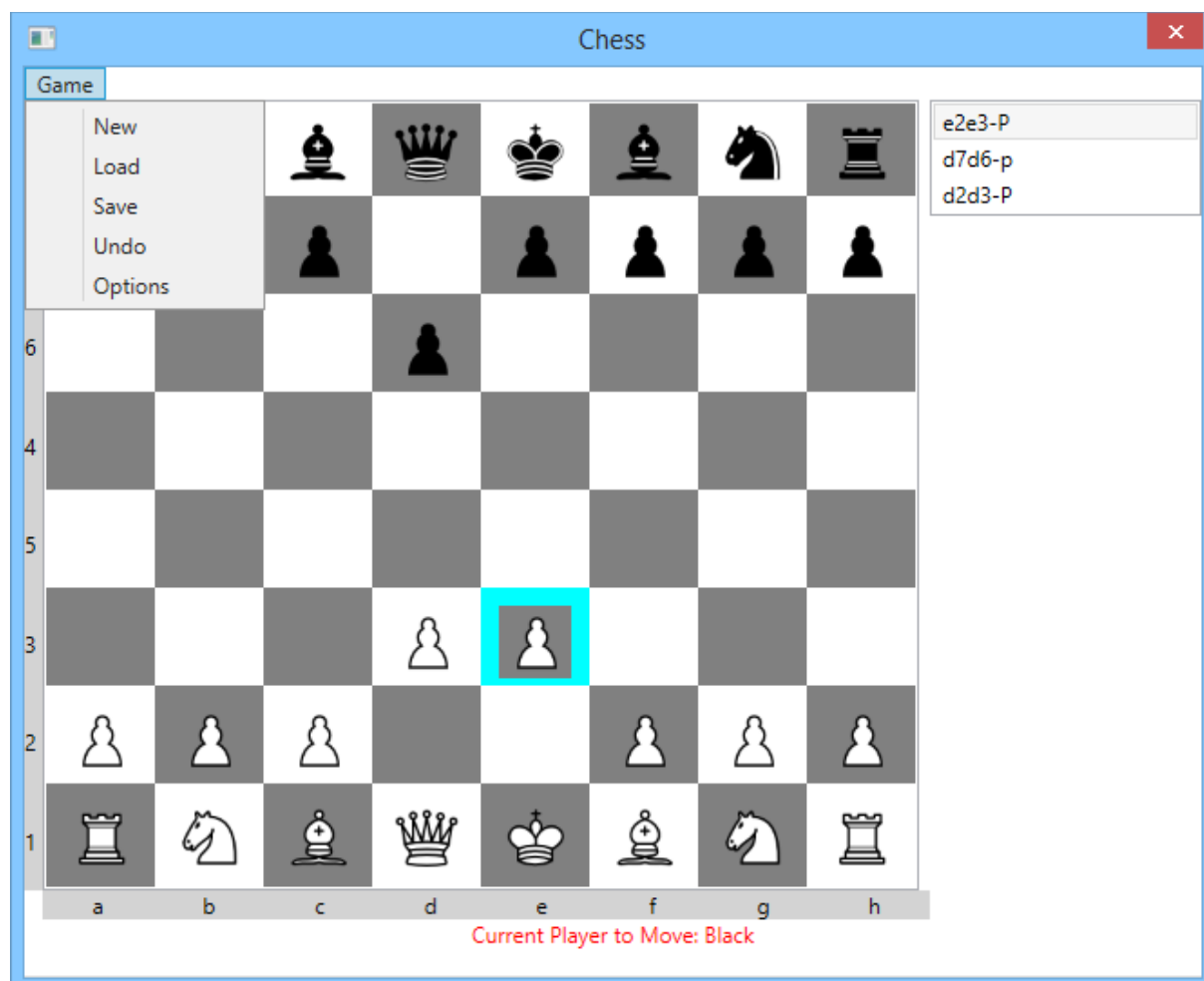


Figura 11
Ecranul de joc

Utilizatorul poate alege piesa pe care dorește să o mute. Această acțiune va duce la colorarea pătratelor pe care piesa poate ajunge, iar pătratele ce reprezintă un atac sunt colorate diferit (figura 12). Pentru a ajuta utilizatorul să observe ultima mutare făcută de adversar aceasta este scoasă în evidență colorând pătratul corespunzător cu o culoare deschisă diferită de celelalte.

2. Prezentarea aplicației dezvoltate

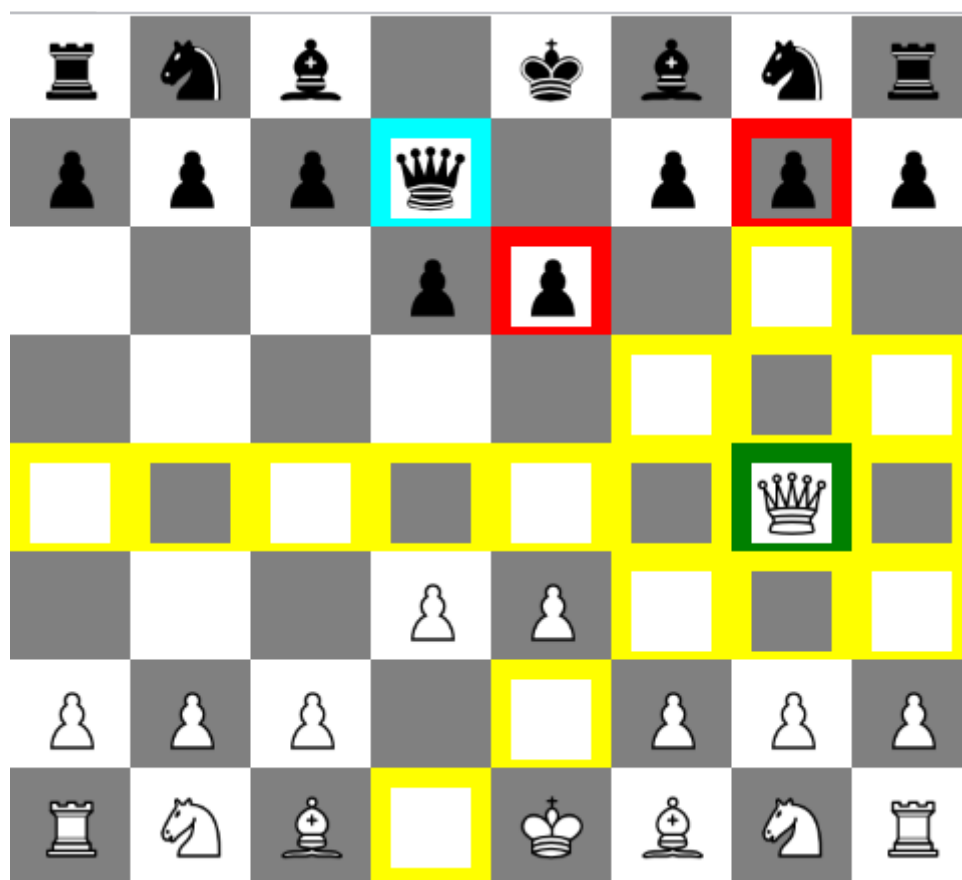


Figura 12

Mutările pe care la poate face regina jucătorului alb

Istoricul poate fi utilizat de către utilizator pentru a marca pe tabla de șah mutările anterioare efectuate (figura 13), această acțiune se face prin a selecta din istoricul cu lista de mutări mutarea ce se dorește a fi analizată, moment în care pe tabla de șah se vor colora pătratele corespunzătoare mutării efectuate.

2. Prezentarea aplicației dezvoltate

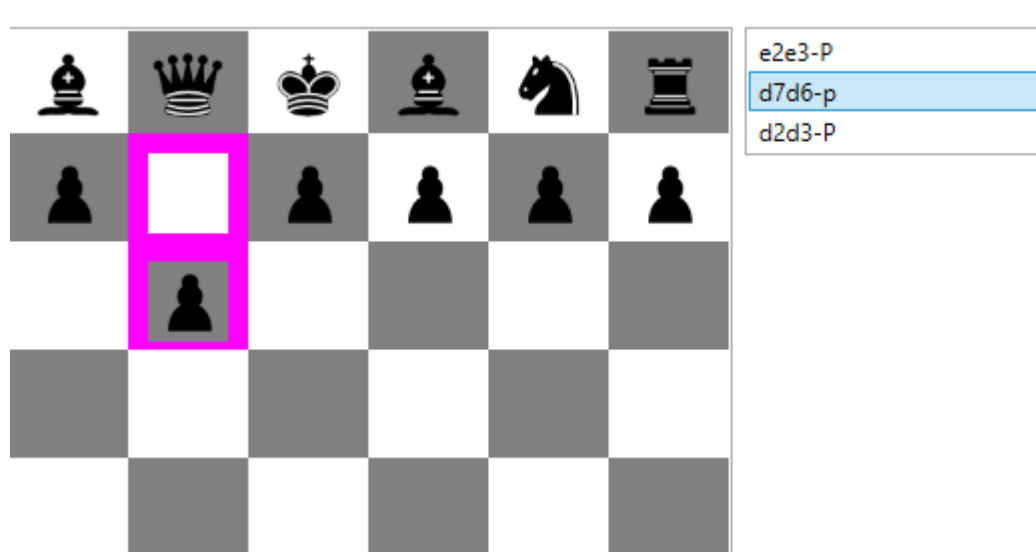


Figura 13

Marcarea unei poziții selectate din istoric

Pentru reprezentarea elementelor grafice se folosesc componentele logice care au responsabilitatea de a-și descrie starea și a notifica interfața grafică în momentul în care aceasta trebuie să se actualizeze. Șablonul de proiectare Model-View-ViewModel presupune că Modelul și ViewModelul sunt responsabile pentru a reține starea în care se află sistemul și de a notifica interfața în momentul în care se produce o schimbare ce este relevantă pentru utilizator. Astfel ViewModel preia informațiile necesare din Model și notifică interfața să se actualizeze corespunzător. Modelul conține logica sistemului: calculează pătratele ce trebuie desenate, reține informații despre mutările anterioare, calculează rândului jucătorului care trebuie să mute și toate operațiile necesare jocului. În continuare vor fi descrise diagramele de clase al modelului jocului de șah.

2. Prezentarea aplicației dezvoltate

2.2.2.1 Diagrame de clasă

În acest capitol vor fi descrise diagramele de clasă care au rolul de a prezenta clasele sistemului, împreună cu atributele, operațiile și relațiile existente. Prima diagramă de clasă este cea a modelului ce descrie o piesă de șah împreună cu strategia de mutare asociată cu aceasta (figura 14).

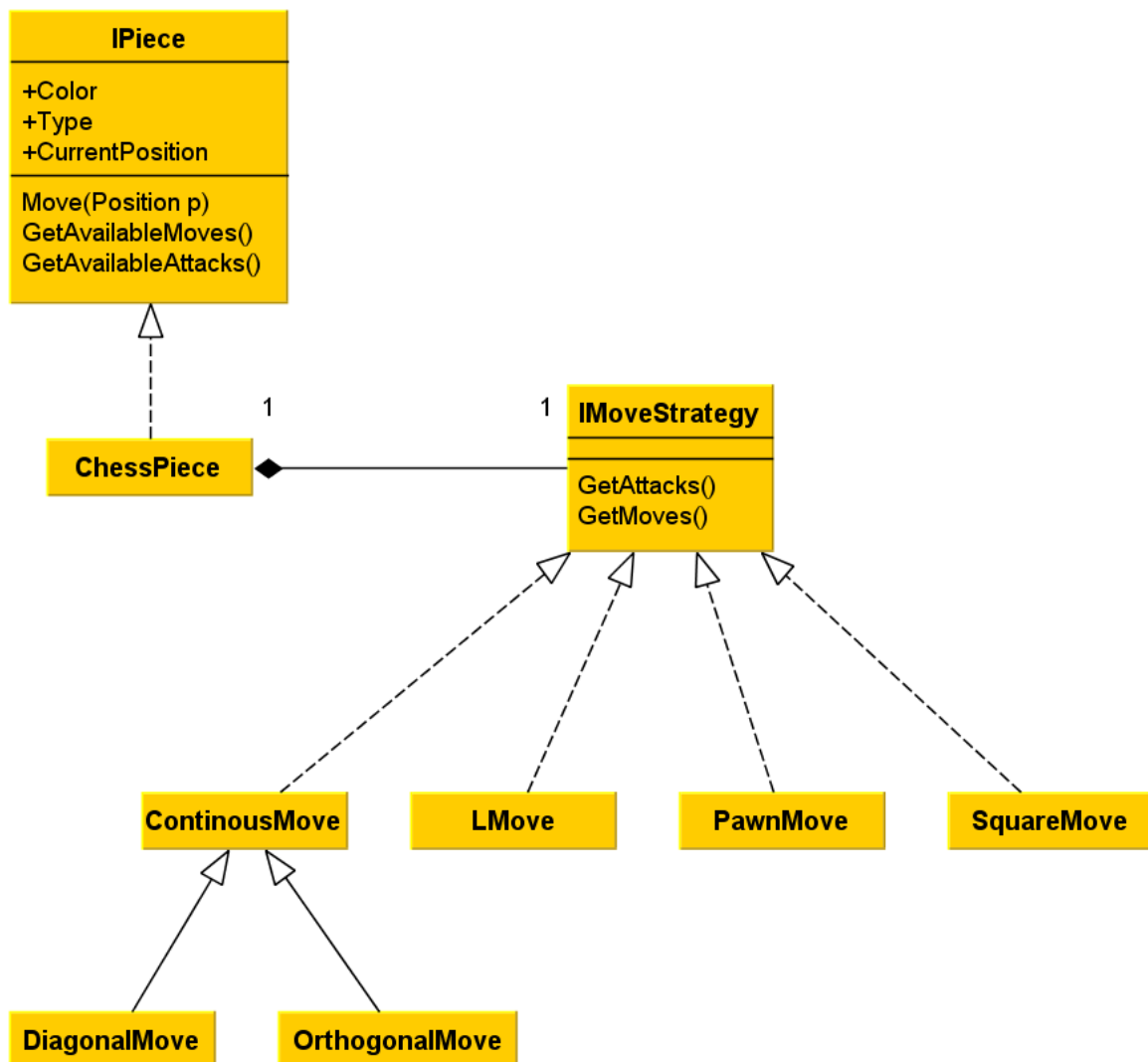


Figura 14
Diagrama de clasă a modelului piesei de șah

2. Prezentarea aplicației dezvoltate

În diagrama de deasupra fiecare piesă de șah este descrisă folosind culoare, tipul și poziția curentă pe care le are împreună cu o strategie de mutare ce îi definește comportamentul ca și piesă al cărei tip este asociat. În cazul reginei se folosește o combinație de două strategii de mutare: abilitatea de a se mișca pe diagonală al nebunului și caracteristica turei de a face mișcări ortogonale în linie dreaptă. Strategiile de mutare au responsabilitatea de a genera mișcări de mutare și de atac pe baza poziției curente al piesei de care aparțin. Există câte o strategie de mutare pentru fiecare tip de mișcare posibilă, astfel avem: mișcarea cu o unitate sau două a pionului în funcție de poziția de start, mișcarea în L a calului, mișcarea cu o unitate în jur necesară pentru rege, două tipuri de mișcări continue, una în diagonală pentru nebun și una în linie dreaptă pentru tură, iar mișcare pentru regină folosește o combinație al mișcării în diagonală și al mișcării în linie dreaptă. În continuare va fi descrisă diagrama de clasă al unui jucător, este natural ca un jucător să conțină o listă de piese și informații relevante despre jucător, precum și acțiunea de a muta o piesă(figura 15).

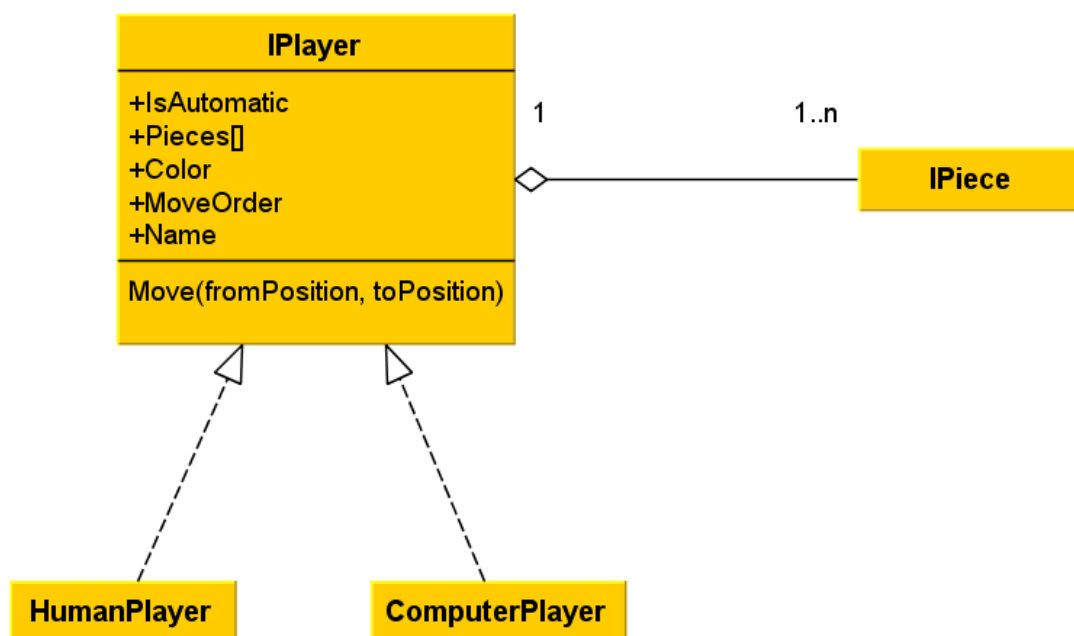


Figura 15

Diagrama de clasă al modelului jucătorului

2. Prezentarea aplicației dezvoltate

După cum observăm jucătorul este generalizat în două clase: una pentru calculator și una pentru jucătorul uman. Diferența dintre cei doi jucători constă în faptul că jucătorul uman trebuie să facă o acțiune asupra interfeței pentru a muta o piesă, iar jucătorul artificial, calculatorul, oferă o acțiune în mod automat ca și răspuns la configurația unei table de șah când îi survine rândul de a muta o piesă, nefiind necesară interacțiunea aplicației cu utilizatorul. Punctul central al aplicației este reprezentat de modelul tablei de joc, acest model deține regulile de joc, informații despre starea tablei, jucătorul al cărui rând este să mute și interpretează comenzile primite de la utilizator. Modelul tablei de joc este ilustrat în figura de mai jos (figura 16).

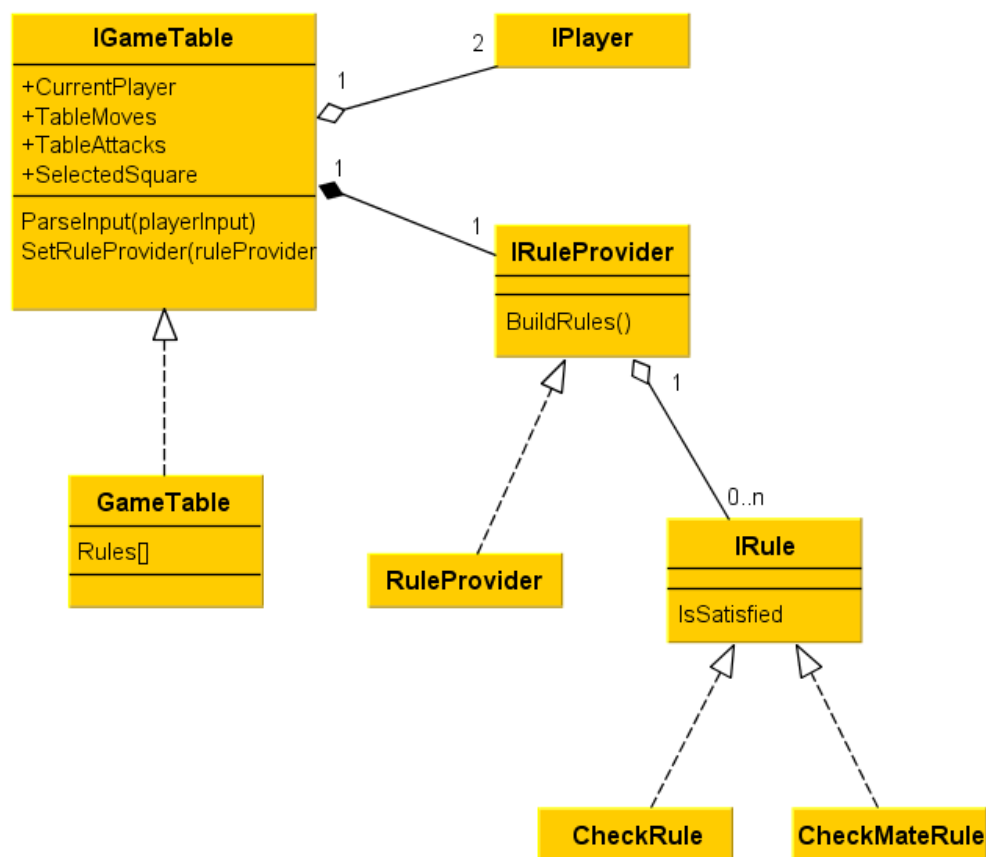


Figura 16

Diagrama de clasă al modelului tablei de joc

2. Prezentarea aplicației dezvoltate

Modelul tablei de joc folosește un furnizor de reguli pentru a verifica pozițiile de șah și de șah mat, acestea sunt executate după fiecare mutare a jucătorului, iar în cazul în care o regulă este satisfăcută asta presupune îndeplinirea condiției de șah/șah mat și jucătorul trebuie anunțat, respectiv mutarea trebuie anulată. Informațiile legate de starea jocului: pătratele pe care o piesă se poate deplasa, pătratele pe care o piesă le poate ataca, lista cu mutările efectuate(pentru a permite anularea lor), pătratul selectat sunt păstrate de către modelul tablei de șah. Stabilirea următorului jucător este responsabilitatea modelului tablei de șah, după fiecare mutare executată cu succes de către un jucător se schimbă rândul jucătorului curent. Tabla de joc necesită piesele tablei împreună cu jucătorii pentru a putea începe meciul, această responsabilitate de furnizare a pieselor aparține unei clase ce folosește șablonul de proiectare fabrică(figura 17). Piesele sunt ulterior distribuite jucătorilor corespunzători, în funcție de culoare.

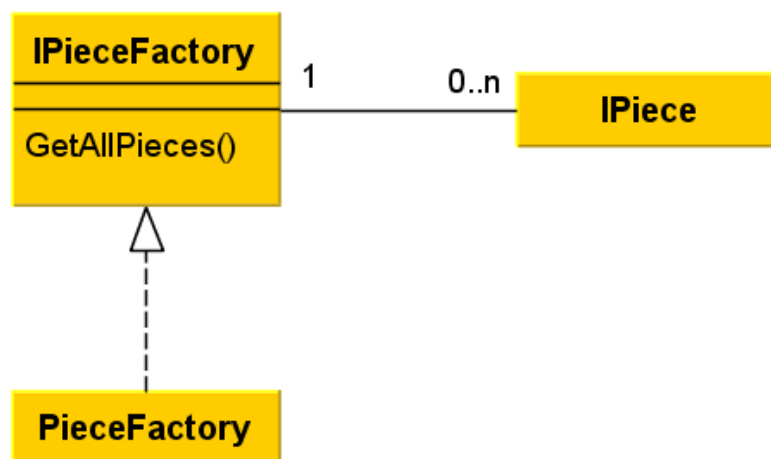


Figura 17

Diagrama de clasă al furnizorului de piese

2.2.3 Serverul: gazda motorului de șah

Fiind dezvoltat în limbajul de programare C, motorul de șah trebuie importat

2. Prezentarea aplicației dezvoltate

într-o librărie .NET de unde va putea fi apelat de o altă aplicație ce dorește a folosi metodele disponibile. Pentru a putea importa librăria în .NET proiectul motorului de șah trebuie compilat folosind setările necesare astfel încât rezultatul compilării să fie o „dynamic-link library”, implementarea Microsoft al conceputului de librărie ce poate partajată. După importarea librărie într-un proiect specific .NET se pot expune metodele necesare. Această librărie are rolul de a accesa metodele dorite, este practic doar un intermediar. În scopul reutilizării motorului de șah, librăria .NET nou creată este găzduită și apelată de către un serviciu web. Serviciul este de tip REST (Representation State Transfer) și oferă metode de tip GET ce pot fi apelate folosind protocolul HTTP pentru a utiliza funcționalitățile motorului de șah(figura 18). Motivația folosirii acestei arhitecturi de structurare a serverului ce găzduiește motorul de șah este justificată prin faptul că se urmărește lipsa unei dependențe directe al clientului față de librăriile ce conțin implementarea motorului de șah. Astfel clientul și serviciul web se pot afla în locații diferite, singura condiție este să existe o rețea prin care să poată comunica. Avantajul este faptul că operația ce solicită puterea de calcul al procesorului este efectuată de către server ceea ce este un lucru important mai ales în cazul dispozitivelor mobile sau cu putere slabă de calcul.

2. Prezentarea aplicației dezvoltate

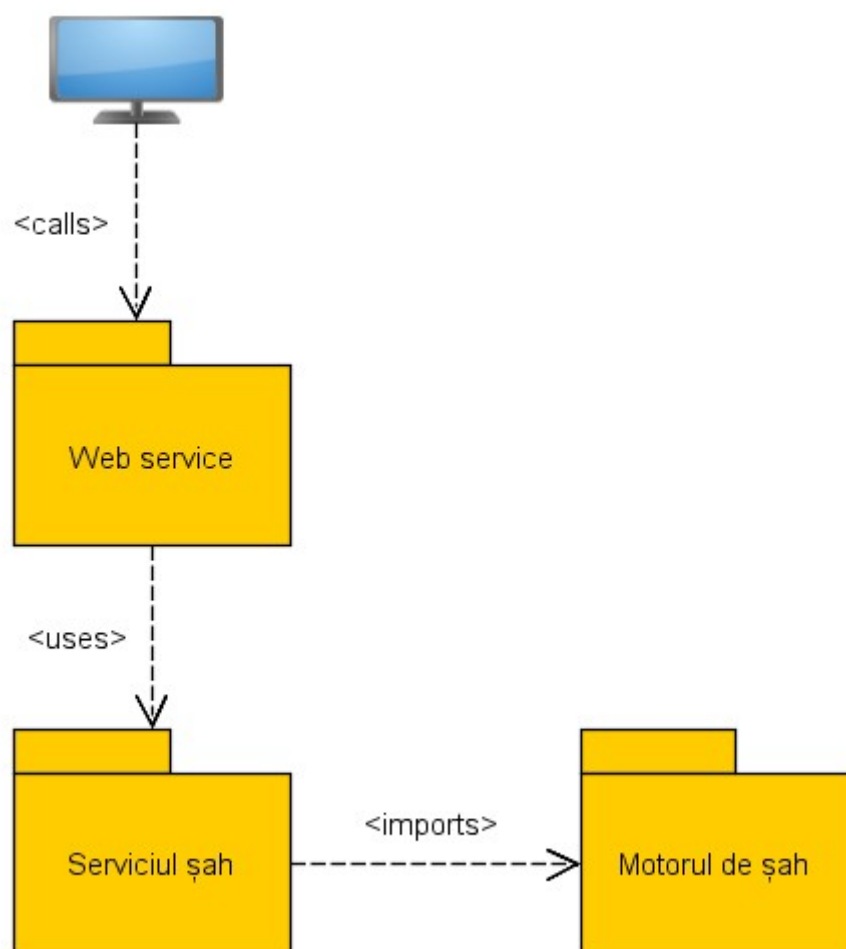


Figura 18

Diagrama de pachete al componentelor serverului