Chess Game

18.01.2021

Îndrumător:

dr. ing. Daniel Morariu

Student:

Tiboldi Roberto-Ciprian

(224/1)

Istoric Versiuni

Data	Versiune	Descriere	Autor
<01/12/2020>	1.0	In aceasta versiune avem 2 ferestre principale, cea de început si tabla. Sunt prezente 2 clase: Board, si Piece. Am creat metodele pentru desenarea tablei si a pieselor.	Tiboldi Roberto
<03/12/2020>	2.0	In acesta versiune s-a extins funcționalitatea clasei Board, astfel s-a poziționat fiecare piesa pe tabla. Apoi am adăugat 2 metode pentru selectarea si mutarea unei piese de pe tabla.	Tiboldi Roberto
<06/12/2020>	3.0	Mutarea pieselor a solicitat prezenta unei metode care sa valideze mutarea, care de asemenea a fost adăugată într-o prima varianta.	Tiboldi Roberto
<03/01/2021>	4.0	In aceasta versiune a fost adăugată partea de rețea. Aceasta a permis conexiunea celor doua forme respectiv Clientul si Serverul.	Tiboldi Roberto
<07/01/2021>	5.0	In aceasta versiune s-a lucrat la implementarea transferului de date cu ajutorul rețelei.	Tiboldi Roberto
<10/01/2021>	5.2	In aceasta versiune au fost adăugate unele funcții care se ocupa de regulile de baza ale jocului precum: „ChangeTurn()”, „RecentMove()” si „EndGame” .	Tiboldi Roberto
<12/01/2021>	5.3	In aceasta versiune a fost actualizat sistemul de validare a mutărilor. De asemenea a fost implementata o clasa pentru fiecare piesa.	Tiboldi Roberto
<14/01/2021>	5.4	In aceasta versiune a fost finalizat transferul de date prin rețea. Apoi a fost adăugată o noua clasa denumita EmptyPiece pentru a marca o poziție in care nu se afla nicio piesa. A fost introdus si o modalitate de a termina jocul (Easy Checkmate).	Tiboldi Roberto

Cuprins

1 Specificarea cerințelor software

1.1 Introducere

In cadrul acestui proiect sa urmărit crearea unui joc de Șah in rețea, care sa permită transmisia, respectiv conexiunea intre un Client si un Server in vederea desfășurări partidei de Șah.
Proiectul este alcătuit dintr-un fereastra principala, unde se introduc datele de conectare, după conexiune, apare într-o alta fereastra de dialog tabla de Șah si piesele. Astfel jocul poate sa înceapă. Jocul se sfârșește atunci când unul dintre cei doi jucători își pierde regele.

1.1.1 Obiective

Principalele obiective ale proiectului:

- Crearea conexiuni intre Client si Server ✓
- Afișarea tablei de joc si a pieselor ✓
- Posibilitatea de a selecta piesa dorita ×
- Posibilitatea de a muta piesa selectata ✓
- Mutarea pieselor conform regulilor de joc ✓
- Rotirea tablei de joc. ✓
- Transmisia de date si actualizarea permanenta a jocului. ✓
- Terminarea jocului si închiderea canalului de conexiune. ✓
- Modificarea designului ferestrelor. ×

1.1.2 Definiţii, Acronime şi Abrevieri

Pe parcursul dezvoltării programului s-au folosit denumiri cat mai sugestive, care sa reflecte funcționalitatea acelor obiecte, clase, membre sau metode. De exemplu obiectul CurrentPiece din clasa Piece reprezintă piesa curenta, metodele SelectPiece() si MovePiece() se ocupa de selectarea si mutarea unei piese pe tabla.

Pentru ferestrele programului sa ales numele MainForm care reprezintă forma principala unde are loc conexiunea, si pentru cea de a doua forma avem numele ChessBoardUI deoarece aceasta afișează tabla, piesele si actualizările de pe parcursul jocului.

Clasele au fost denumite in funcție de utilizarea lor. Clasele principale fiind Board si Piece. Celelalte clase fiind adăugate ulterior pentru a facilita implementarea unei metode de validare a mutări care sa fie specifica fiecărei piesa in parte. Aceste clase de pe urma având numele piesei la care fac referința: EmptyPiece, Pawn, Rook, Bishop, Knight, Queen si King.

1.1.3 Tehnologiile utilizate

Imaginile pieselor au fost descărcate de pe un website menționat in bibliografie, si apoi au fost redimensionate cu ajutorul unui alt website.

Programul Excel a fost folosit pentru gândirea si organizarea claselor si de asemenea sa folosit VisualStudio2019 pentru implementarea codului.

1.2 Cerințe specifice

Funcționalitățile aplicației care au fost realizate sunt:

1. Afișarea pieselor pe tabla de joc.
2. Selectarea si mutarea pieselor.
3. Codificare si Decodificarea mesajelor transmise si recepționate pe rețea.
4. Validarea mutări.

2 Funcționalitate 2

2.1 Descriere

O prima funcționalitate de mare importanta pentru aceasta aplicație este „Selectarea si mutarea pieselor”. Aceasta funcționalitate este alcătuita din doua parți, prima parte permite selectarea piesei si cea de a doua permite mutarea piesei respective.

2.2 Fluxul de evenimente

2.2.1 Fluxul de bază

La baza acestui sistem de selecție si mutare se afla doua funcții care acționează asupra vectorul pieces[] de tip Piece din clasa Board. In clasa Board practic se ordonează piesele pe tabla, acestea fiind in cele din urma desenate cu ajutorul funcționalității : „Afișarea pieselor pe tabla de joc”.

In primul rând obiectul „pictureBox” (tabla de joc) înregistrează poziția cursorului atunci când utilizatorul se folosește de „Left-Click” pentru a selecta o piesa sau o pozitie.

Funcția „pictureBox1_MouseDown” primește un MouseEventArgs, notat e, cu ajutorul acestui eveniment sunt salvate coordonatele poziție cursorului care apoi permite determinarea pătratului in care se afla piesa selectata. Apoi are loc desenarea tablei pentru a marca selecția.

In al doilea rând ne folosim tot de funcția „pictureBox1_MouseDown” pentru a primi poziția in care utilizatorul dorește sa mute piesa selectata care acuma este salvata in obiectul „CurrentPiece”. Cu ajutorul acestui obiect si cu coordonatele primite, se șterge piesa din poziția anterioara, se muta in noua poziție si se redesenează tabla astfel încât sa fie vizibila mutarea.

2.2.2 Pre-condiții

Pentru ca aceasta funcționalitate sa aibă loc, in primul rând utilizatorul trebuie sa se asigure ca sa creat conexiunea intre Client si Server, acest lucru fiind evidențiat de apariția tablei de joc.

In al doilea rând daca este rândul lui, acest lucru fiind de asemenea specificat in „InfoBox-ul” pe care îl are in partea de jos a ecranului, poate sa selecteze ce piesa dorește si sa o mute in poziția corespunzătoare.

2.2.3 Post-condiții

In urma selectări unei piese, utilizatorul poate observa marcarea cu culoare roșie a pătratului care conține piesa respectiva. Apoi daca utilizatorul selectează poziția in care dorește sa mute piesa, după verificarea mișcării de către funcționalitatea de validare, piesa respectiva o sa fie mutata in poziția dorita.

3 Funcționalitate 3

3.1 Descriere

O alata funcționalitate fundamentala pentru aceasta aplicație este : „Codificare si Decodificarea mesajelor transmise si recepționate pe rețea”. Aceasta fiind foarte utila in stabilirea conexiuni intre cei doi jucători si in actualizarea tablei de joc.

3.2 Fluxul de evenimente

3.2.1 Fluxul de bază

I PARTE

Pentru realizarea acestei conexiuni sa creat un canal de transfer a informației cu ajutorul clasei „NetworkStream” din biblioteca „System.Net.Sockets;”. Apoi pentru codificarea mesajului sa folosit codificarea in ASCII cu ajutorul bibliotecii: „System.Text.Encoding.ASCII”.

Pentru citirea si scrierea mesajului s-au folosit funcțiile WriteMessage() care codifica un String de date in un sir de biți, si ReadMessage() care decodifica șirul de biți într-un String.

```
private String ReadMessage()
        {
            Byte[] bytes = new Byte[256];
            String data = null;
            Int32 i = stream.Read(bytes, 0, bytes.Length);
            data = System.Text.Encoding.ASCII.GetString(bytes, 0, i);
            return data;
        }
private void WriteMessage(String data)
        {
            Byte[] bytes = new Byte[256];
            bytes = System.Text.Encoding.ASCII.GetBytes(data);
            stream.Write(bytes, 0, bytes.Length);
               }
```

Apoi când jucătorul care vrea sa fie Host apasă pe butonul „Host” se creează serverul care așteaptă sa primească o cerere de la client cu versiunea jocului, daca versiunea e corecta trimite mesajul „Connected” si începe meciul altfel trimite mesajul „Wrong version”.

Celălalt jucător apasă pe butonul de „Play” astfel se creează Clientul care trimite mesajul cu versiunea si așteaptă sa primească confirmarea, apoi jocul poate sa înceapă.

A II-a PARTE

După ce jocul începe si are loc prima mutare, informațiile relative schimbărilor care au loc pe tabla de joc a primului jucător trebuie sa apară si pe tabla de joc a celui de al doilea jucător, astfel cu ajutorul funcției WriteMessage anunțam schimbarea si trimitem indecși care au fost afectați.

In timp ce primul jucător muta piesa cel de al doilea aștepta sa primească răspunsul, aceasta sarcina este efectuata de funcția WaitForResponse() care după ce a primit informațiile necesare le decodifica si le transmite către funcția UpdateGame().

```
private async void WaitForResponse()
        {
            try
            {
                Byte[] bytes = new Byte[256];
                String data = null;
```

```
            Int32 i = await stream.ReadAsync(bytes, 0, bytes.Length);
            data = System.Text.Encoding.ASCII.GetString(bytes, 0, i);
            UpdateGame(data);
        }
        catch (ObjectDisposedException ex)
        {
            InfoBox.Text = "Opponent disconnected. You won!";
            CurrentGame.EndGame();
            stream.Close();
        }
    }
```

3.2.2 Pre-condiţii

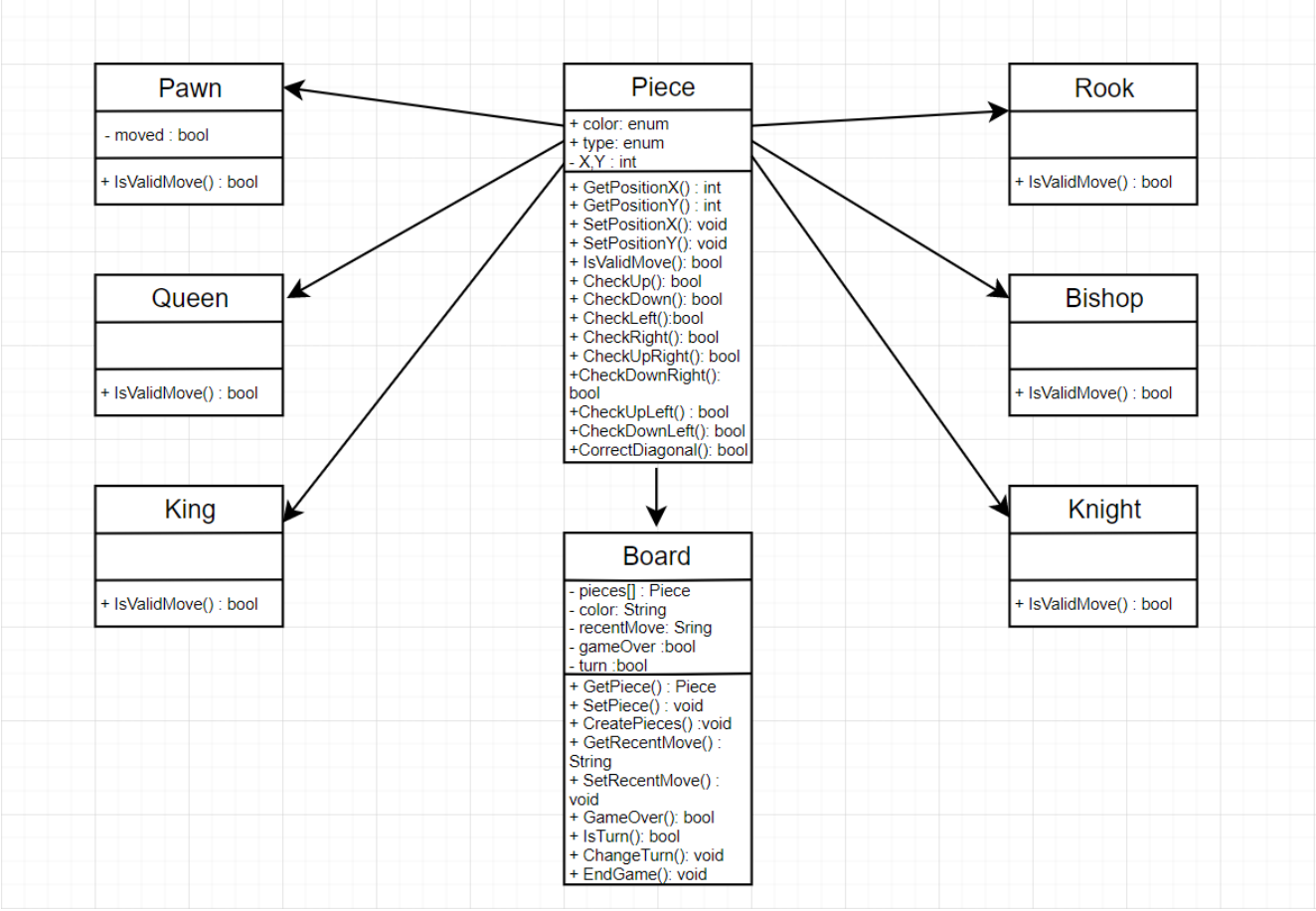
Pentru ca jocul sa înceapă, in primul rând trebuie sa se stabilească conexiunea intre cei doi jucători, acest lucru are loc daca ambii utilizatori au aceeași versiune a jocului. Apoi unul dintre ei trebuie sa insereze un Ip si sa devina Host, prin apăsarea butonului respectiv. Acuma cel de al doilea jucător poate sa inserează Ipul si sa apese butonul de „Play”.

3.2.3 Post-condiţii

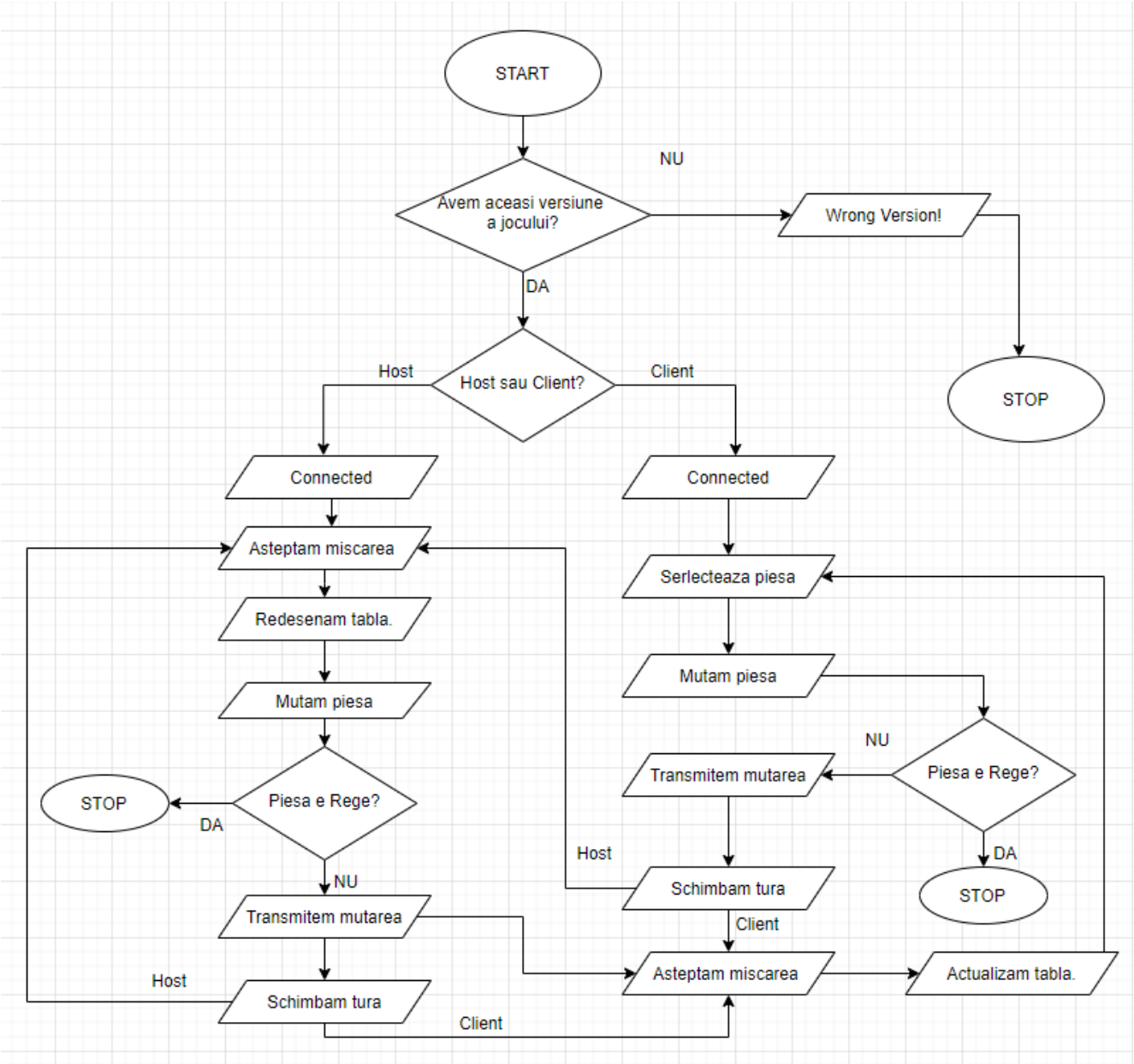
In urma inserării Ipului si apăsări butoanelor respective, se va putea observa ca jocul a început, deoarece apare tabla de joc si piesele respective.

4 Implementare

4.1 Diagrama de clase



4.2 Descriere detaliată



5 Bibliografie

stackoverflow.com

geeksforgeeks.com

microsoft.com

www.adamberent.com

https://docs.microsoft.com/en-us/dotnet/api/system.drawing.bitmap?view=dotnet-plat-ext-5.0

https://docs.microsoft.com/en-us/dotnet/api/system.collections.generic.dictionary-2?view=net-5.0

https://docs.microsoft.com/en-us/dotnet/standard/base-types/character-encoding

https://www.adamberent.com/wp-content/uploads/2019/02/GuideToProgrammingChessEngine.html#_Toc465072613

https://www.geeksforgeeks.org/design-a-chess-game/

Online Tools:

https://onlinepngtools.com/resize-png