NU TE SUPARA FRATE

16 IANUARIE 2025

Indrumator: Student:

Dr. Ing. Daniel Morariu Bogorodea Stefan-Antonio

Grupa 22/1 TI

**ISTORIC VERSIUNI**

|  |  |  |  |
| --- | --- | --- | --- |
| **DATA** | **VERSIUNE** | **DESCRIERE** | **AUTOR** |
| 1.12.2024 | 1.1 |  | Bogorodea Stefan  Antonio |
| 4.1.2025 | 1.2 | Am reusit sa rezolv o problema prin care imi ramane fereastra de care am nevoie activa chiar si cand apas pe un buton gresit sau un buton care declanseaza o erroare. Arata astfel: | ChatGPT |
| 7.1.2025 | 1.3 | Am rezolvat o problema prin care nu defineam bine un shape sau un label, informatia am gasit-o pe un site de specialitate: | StackOverflow |
| 12.1.2025 | 1.4 | A fost rezolvata o problemă în care pionii nu reușeau să treacă de un obiect (shape) din cauza unei valori incorecte a poziției acestora. | ChatGPT |
| 15.1.2025 | 1.5 |  | Bogorodea Stefan  Antonio |

Avem, in primul rand, prima versiune a aplicatiei, cea in care am creat cele 2 clase si logica de mutare (pentru fiecare jucator in parte), dupa care am avansat si am dat de o problema in care eu aveam nevoie ca, si dupa apasarea unui buton din Form, fereastra sa ramana inca activa din diferite considerente (ex: aveam un pion al jucatorului curent pe acea pozitie si trebuia sa selectez alt pion pentru a fi mutat). Dupa care, in versiunea 1.3, am dat de o alta problema,legata de adaugarea formelor intr-un vector de shape-uri, nu stiam exact de folosirea „nullptr”, initializand shape-urile direct cu numere, lucru care nu functiona si nu ma ajuta in dezvoltarea aplicatiei. In versiunea 1.4 a aplicatiei, am intampinat o problema care nu imi permitea sa mut pionii mai departe de pozitia 1 in momentul in care trimiteam pionul serverului catre client si viceversa, asa ca am gasit metoda prezentata in poza de mai sus pentru parcurgerea intregii table de joc, iar in final, avem versiunea prezentata astazi.

**CUPRINS**

**ISTORIC VERSIUNI 2**

**CUPRINS 4**

**1 SPECIFICAREA CERINTELOR SOFTWARE 6**

**1.1 Introducere 6**

1.1.1 Obiective **6**

1.1.2Definitii, Acronime si Abrevieri  **6**

1.1.3 Tehnologiile Utilizate **7**

**1.2 Cerinte Specifice 7**

**2 Mutarea pionilor 8**

**2.1 Descriere 8**

**2.2 Fluxul de evenimente 8**

2.2.1 Miscarea pionilor **8**

2.2.2 Pre-Conditii **9**

2.2.3 Post-Conditii **9**

**2 Transmiterea mesajelor 9**

**2.1 Descriere 10**

**2.2 Fluxul de evenimente 10**

2.2.1 Codificarea si decodificarea **10**

2.2.2 Pre-Conditii **12**

2.2.3 Post-Conditii **12**

**3 IMPLEMENTARE 13**

**3.1 Diagrama de clase 13**

**3.2 Descriere detaliata 14**

**4 BIBLIOGRAFIE 15**

**1 Specificarea cerintelor software**

***1.1 Introducere***

„Nu te supăra, frate!” este un joc de masa clasic pentru 2-4 jucatori, care se joaca cu un zar si 4 pioni pentru fiecare jucator. Scopul jocului este sa-ti duci pionii pe traseul de pe tabla de joc si sa ajungi primul cu toti pionii acasa, inaintea celorlalti.

**1.1.1 Obiective**

Cand am ales tema, m-am gandit din start la anumite obiective pe care vreau sa le indeplinesc, printre ele numarandu-se creearea unei table de joc care sa fie cat mai apropiata de cea reala, lucru care cred ca mi-a si iesit, mutarea pionilor prin intermediul schimbarii culorii shape-urilor, iar la fel ca si mutarea si extragerea din casa a pionilor, tot prin schimbare de culoare a shape-ului a fost facuta. Pe langa asta, am avut ca obiectiv crearea jocului in retea pentru a se putea juca de mai multe calculatoare, obiectiv aproape atins. Consider ca acest obiectiv mai trebuie atins pentru ca jocul sa fie complet. In momentul de fata nu am reusit sa-l indeplinesc complet, am creat reteaua, mutarea jucatorilor pe Server si Client functioneaza, insa cand se ajunge la introducerea pionilor in casa pentru Client si Server, acest lucru nu functioneaza, desi individual am reusit sa fac acest lucru.

**1.1.2 Definitii, Acronime si Abrevieri**

In acest proiect am folosit in mare parte definitii ale variabilelor, usor de inteles, care specifica exact despre ce este vorba, cum ar fi: PionVerdeX, PionGalbenX (X reprezentand numarul pionului (de la 1 la 4)), zarButton, mutaButton, PozitiaX (X reprezentand un numar pentru fiecare Shape si tot o data indica si pozitia pionului pe tabla de joc), insa pe langa acestea am folosit si definitii si abrevieri poate mai greu de inteles de catre utilizator:

-vechi1, vechi2, vechi3, vechi4: este folosit pentru a prelua o valoare compusa din mai multe valori adunate si pentru a fi trimisa spre client/server pentru mutarea pionului.

-pasiRamasi1, pasiRamasi2, pasiRamasi3, pasiRamasi4: folosite pentru a sti cate pozitii poti inainta in casa in cazul in care esti in dreptul acesteia.

-pozitieInitialaPionGalben1, 2, 3, 4: este folosita pentru a sti de unde pornim exact cu pionul in momentul scoaterii acestuia din casa, iar pe langa asta, ne ofera si un punct de start pentru noua pozitie si pentru variabila pozitieNoua1, 2, 3, 4 care creste in functie de pozitieInitialaPionGalben1, 2, 3, 4 si valoarea de pe zar.

-variabila “start”: ne ajuta sa transmitem o valoare intre server si client care ne ajuta la logica butoanelor de zar la inceputul jocului.

-pion1Scos, pion2Scos, pion3Scos, pion4Scos: sunt variabile booleene care ne ajuta sa stim daca un pion este scos sau nu din casa, iar ca in cazul variabilelor de mai sus, si pion1Casa, pion2Casa, pion3Casa, pion4Casa sunt tot variabile booleene, doar ca acestea ne ajuta sa stim daca pionii au ajuns in casa sau nu.

**1.1.3 Tehnologii Utilizate**

In cadrul proiectului nu au fost folosite technologii externe, doar clientul software Embarcadero C++ Builder 12, documentatia de la curs, diferite siteuri de specialitate (Ex: Stack Overflow, YouTube)

***1.2 Cerinte Specifice***

- Aruncarea zarului: una dintre cele mai importante functii ale jocului este aruncarea zarului, jocul functioneaza pe baza valorilor date de zar. Extragerea pionilor din casa, mutarea lor, verificarea anumitor conditii, toate sunt legate de valoarea zarului de la apasarea butonului.

- Scoaterea la joc a pionilor: aceasta functionalitate functioneaza prin intermediul schimbarii culorii shape-ului si prin intermediul valorii zarului, in momentul in care se arunca zarul, iar valoarea acestuia este 6, aceasta functie se activeaza si permite jucatorului scoaterea la joc a pionului.

- Mutarea pionului: aceasta functie permite jucatorului sa mute pionii atat in Server, cat si in Client, mutarea se face prin schimbarea culorii shape-ului.

- Verificarea unei pozitii: functia se refera la verificarea pozitiei urmatoare pe care urmeaza sa ajunga un pion, impiedicand mutarea in cazul in care un pion aliat este pe acea pozitie si redirectionarea spre mutarea altui pion valabil pe tabla.

- Trimiterea prin socketuri a informatiilor pentru a fi prelucrate de Server/Client: aceasta functie este creata pentru a putea trimite informatii intre client si server, avand in vedere ca mutarea pionilor pe tabla de joc a jucatorului advers se face prin mesaje codificate, putem spune ca este una dintre cele mai importante functii din proiect, mai exact putem da un exemplu: la mutarea unui pion, serverul trimite catre client urmatorul mesaj „M P2 12 16”, clientul primeste mesajul si il descifreaza, el intelegand de fapt asta: Mut pionul 2 de la pozitia 12 la pozitia 16.

**2 Mutarea pioniilor**

***2.1 Descriere***

Consider aceasta functie cea mai importanta deoarece, prin intermediul ei, putem spune ca „jocul are viata”. Intr-un joc de nu te supara frate, mutarea pionilor este cea mai importanta, iar mutarea in cadrul acestei aplicatii a fost implementata, dupa cum am spus si anterior, prin intermediul schimbarii de culoare a unui Shape.

***2.2 Fluxul de evenimente***

**2.2.1 Miscarea pioniilor**

In primul rand, trebuie sa incepem prin pornirea serverului. La inceput, orice buton de miscare a pionilor va fi oprit pana cand ambii jucatori apasa butonul „START” si jocul incepe. Dupa butonul de start, fiecarui jucator i se activeaza butonul de zar, cel de la care incepe miscarea, si incep sa se verifice valorile. Daca un jucator nu da valoarea 6 la zar si nu are niciun pion scos pe tabla de joc, il va lasa pe urmatorul jucator sa dea cu zarul si tot asa pana cuiva ii revine valoarea 6, iar de aici incepe cu adevarat mutarea pionilor. Odata ce dai valoarea 6, pe langa ca mai esti lasat sa dai inca o data de zar, ti se si deblocheaza 2 butoane: cel de „SCOATE PION” si cel de „MUTA PION”. Acest lucru se intampla doar cand iti pica valoarea 6 la zar; in rest, orice valoare care pica pe zar va activa doar butonul de muta, buton care ne si intereseaza in clipa de fata. Dupa ce am stabilit cum un jucator este lasat sa mute un pion, incepe mecanismul de mutare, totul incepe de la valoarea initiala (valoarea Shape-ului dedicat startului fiecarui jucator). Dupa care, cu cat dai cu zarul, acea valoare creste si ii este oferita butonului care face urmatoarele lucruri: preia valoarea, o introduce in cadrul unei metode de mutare din cadrul clasei „Pozitie”, unde este adunata cu valoarea zarului si este data drept o noua valoare numita „pozitieNoua”, aceasta fiind urmatoarea pozitie pe care urmeaza sa se afle butonul. Dupa care, verifica daca pozitia pionului este intre 25 si 29, in cazul in care un numar prin care poate fi introdus pionul in casa este aratat pe zar, dupa care trecem la mutarea propriu-zisa. Obtinem pozitia veche a Shape-ului si pozitia noua a acestuia prin intermediul urmatoarelor metode:

TShape\* shapeVechi = pozitie.getPozitie(pozitieInitialaPionGalben2);

TShape\* shapeNou = pozitie.getPozitie(pozitieNoua2);

Dupa care verificam daca pe pozitia urmatoare nu cumva exista deja un pion propriu si se trece la mutare prin colorarea Shape-ului vechi in alb, iar a Shape-ului nou in culoarea jucatorului, iar ulterior se transmit aceste lucruri spre Server/Client.

**2.2.2 Pre-conditii**

Pentru a porni aceasta functie de mutare a zarului, in primul rand toti jucatorii trebuie sa apese pe butonul „START”, incepand de la Server, iar la urma Clientul, dupa care se va debloca butonul de zar individual. Dupa cum am prezentat si mai sus, trebuie sa se ajunga la valoarea 6 pe zar pentru a putea scoate pionul, iar dupa ce avem un pion scos pe tabla, valoarea de pe zar nu mai conteaza, butonul de muta deblocandu-se la fiecare apasare de zar. Problema care nu a fost inca rezolvata in cod este aceea ca atunci cand pionul unui jucator pica pe pionul altui jucator, acesta nu trimite pionul advers inapoi in casa, acesta doar se „aseaza” peste el, stricand regulile jocului.

**2.2.3 Post-Conditii**

Dupa rularea metodei de mutare a pionului, jucatorul trebuie sa se astepte sa vada un Shape care contine culoarea pionilor proprii si numele sau in functie de pion, de exemplu: pentru pionul 1, avem „P1”, cum se muta pe urmatoarea pozitie aleasa de zar, stergand vechiul shape, adica colorandu-l in alb.

**2 Transmiterea mesajelor**

***2.1 Descriere***

Avand in vedere ca aceasta functie face parte din mutarea pionilor, consider ca este si ea foarte importanta pentru aplicatie si merita sa fie discutata iar transmiterea de mesaje se face sub forma “M P1 12 16”, care este un mesaj codificat care urmeaza sa fie descifrat de catre Server/Client.

***2.2 Fluxul de evenimente***

**2.2.1 Codificarea si decodificarea mesajului**

Incepem prin a reaminti mesajele in cadrul fiecarei miscari si a le explica dupa care trecem la prezentarea lor in ansamblu. Primul mesaj este del de mutare “M P3 17 21”, dupa care il avem pe cel de scoatere la joc a pionului “C P1 30”, in acest caz, 30 este un numar de sine statator pentru pionul Galben, in cazul pionului verde avem numarul 1 ca si numar de sine statator, iar nu in cele din urma avem mesajul de start care este mai simplu decat cele doua prezentate mai sus, arata astfel: “S 1” sau “S 2”, dictand valoarea care ii va fi atribuita variabilei start din cadrul codului si care ne ajuta la pornirea jocului.

Incepem cu prima, cea de mutare. Este implementata la finalul fiecarui buton de mutare si arata astfel in cadrul clientului:

String mesaj = "M P4 " + IntToStr(vechi4) + " " + IntToStr(pozitieNoua4);

ClientSocket1->Socket->SendText(mesaj);

Dupa cum se observa, trimite o valoare standard dupa care se ghideaza Serverul pentru a sti despre ce mutare este vorba si doua valori convertite la String, ulterior fiind convertite inapoi la Int pentru a putea fi folosite in program de catre server

Funcita din cadrul Serverului pare ceva mai complicata insa nu este deloc asa, aceasta aratand in felul acesta:

String mesaj = "M P4 " + IntToStr(vechi4) + " " + IntToStr(pozitieNoua4);

for (int i = 0; i < ServerSocket1->Socket->ActiveConnections; i++) {

TCustomWinSocket \*clientSocket = ServerSocket1->Socket->Connections[i];

clientSocket->SendText(mesaj);

}

La fel, are aceleasi proprietati ca si cea din client. Trecand la cea de scoatere a pionului din casa este foarte asemanatoare cu cea de mutare, singura diferenta fiind doar mesajul trimis:

String mesaj = "C P3 30";

Aceasta fiind mai simpla, deoarece foloseste direct doua Stringuri si o valoare implicita, iar functia de trimitere a valorii variabilei de start arata la fel de simplu ca cea de extragere a pionului:

String mesaj = "S " + IntToStr(start);

Trecand acum la functionalitate, in fiecare aplicatie, avem cate o metoda de primire si de convertire a mesajului, aceasta metoda primeste in primul rand mesajul dupa care verifica ce mesaj este verificand prima litera din sirul primit:

if (receivedValues.SubString(1, 1) == "M")

Iar dupa ce s-a decis ce despre ce este vorba in mesaj converteste valorile de care are nevoie din String in int, iar fiecare mutare are definita cate o functie speciala in cadrul codului cum ar fi cea de mutare care se aseamana cu ce avem in butonul de muta iar cea de scoatere a pionului asemanandu-se cu butonul de extragere a pionului

**2.2.2 Pre-conditii**

Pentru pornirea acestei functii, vom trece din nou prin pasii pentru activarea butonului de mutare a pionului prezentati la punctul anterior in cadrul „Miscarea pionilor”, aici fiind diferit faptul ca, pe langa jucatorul principal, va fi „afectat” si al 2-lea jucator, acesta va vedea pe propria lui tabla de joc pionii adversarului miscandu-se la pozitiile specificate de primul jucator. O problema pe care am intampinat-o in crearea acestei functii ar fi introducerea in casa a pionului, de data aceasta nu ne referim la eliminarea sa de pe harta, ci doar la introducerea sa in casa de castig a jocului. Chiar daca pe tabla de joc a fiecarui client un pion intra in casa, adversarul tau nu poate vedea acest lucru, doar daca este atent in momentul in care esti aproape de casa cu un pion, el disparand o data cu introducerea lui in casa de jucatorul advers.

**2.2.3 Post-conditii**

Dupa cum am zis si mai sus, pe langa mutarea simpla facuta de fiecare jucator, acum vor fi „afectati” si restul. O data cu apasarea butonului de muta, pe langa ca tu, ca si jucator principal, vei vedea pionul miscandu-se ca la punctul anterior, acum si restul jucatorilor vor vedea pe tabla de joc individuala miscarile pe care le faci si Shape-urile care isi schimba culoarea in functie de culoarea oponentului.

**3 Implementare**

***3.1 Diagrama de clase***

**ZAR**

**# valoare : int**

**+arunca()**

**+Zar()**

**+getValoare()**

**POZITIE**

**# pozitieCurenta : int**

**# labels[40], getLabel(int index) : TLabel\***

**# pozitii[40], getPozitie(int index) : TShape\***

**# labelTextInitial[40] : String**

**+Pozitie()**

**+adaugaPozitie(TShape\* pozitie, int index)**

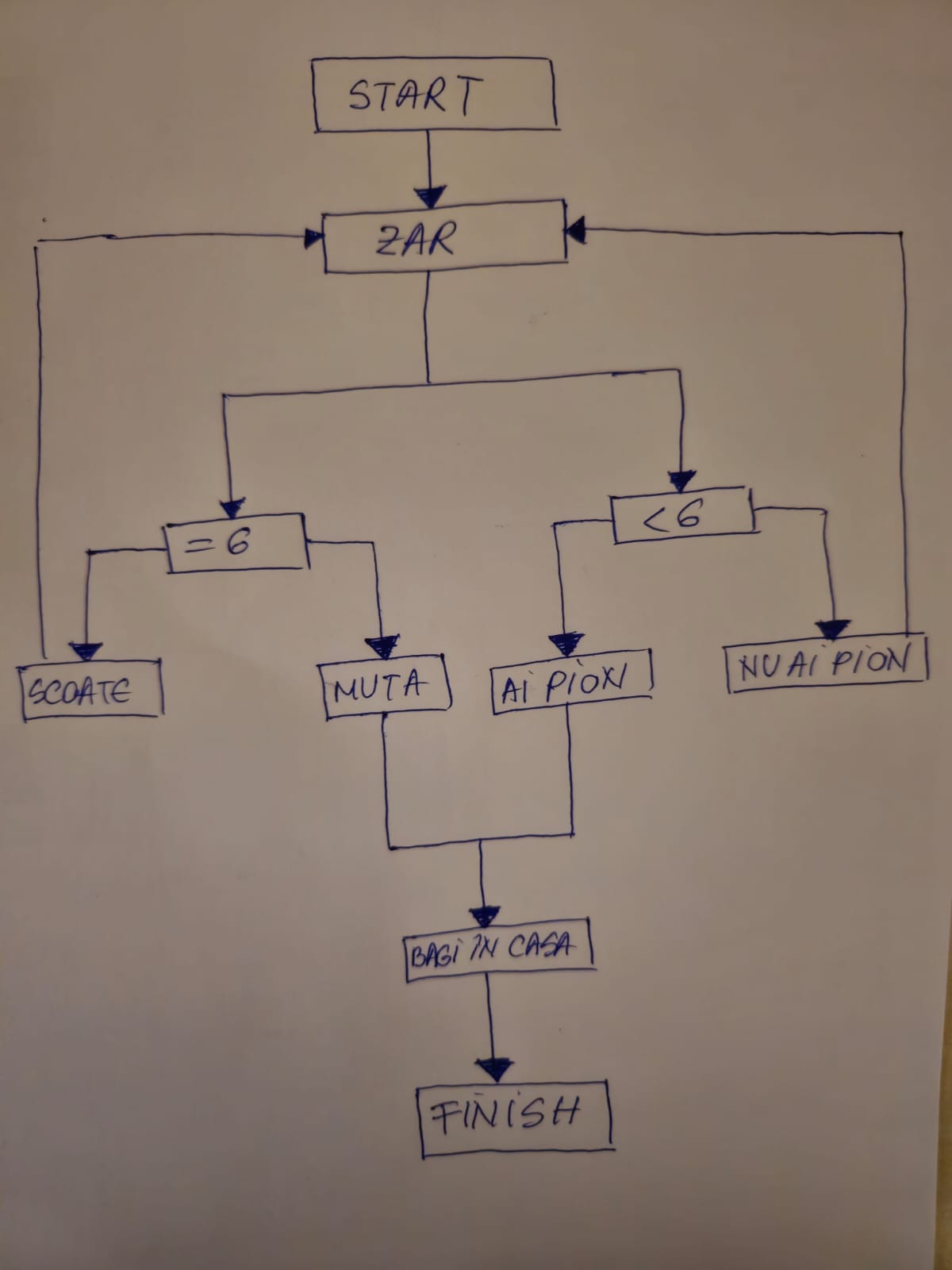
**+adaugaLabel(TLabel\* label, int index)**

**+getZar();**

**+muta(int pozitieInitiala, int valoareZar);**

**+actualizeazaLabel(pozitieVechi, pozitieNou, pionText);**

***3.2 Descriere detaliata***



**4.BIBLIOGRAFIE**

[1] Programare C++ Builder – Windows Controls:

<http://www.functionx.com/cppbuilder/>

[2] Dezvoltare aplicatie cu interfata grafica:

http://docwiki.embarcadero.com/RADStudio/Tokyo/en/VCL

[3] Board-game: “DON’T BE UPSET BROTHER”:

<https://www.youtube.com/watch?v=TZHkiIPFC-I>

[4] Learn C++.org:

https://learncplusplus.org/