ASC-BMP

10.01.2024

Îndrumător:

dr. ing. Daniel Morariu

Student:

Neag Mihai

223/2

Istoric Versiuni

Data	Versiune	Descriere	Autor
06.12.2023	0.1	Creare schelet aplicație	Neag Mihai
09.12.2023	0.2	Implementare funcționalitate fișiere	Neag Mihai
13.12.2023	0.3	Restructurare prelucrare fișier	Daniel Morariu
05.01.2023	0.4	Implementare codificare/decodificare informații	Neag Mihai
07.01.2024	0.5	Implementare mod alternativ de generare a imaginilor	Neag Mihai

Cuprins

1 Specificarea cerințelor software

1.1 Introducere

Aplicația "ASC-BMP" este menită a permite utilizatorului codificarea unui text în cadrul unei imagini bmp și decodificarea ulterioară a acesteia.Astfel îi sunt dispuse două moduri de codificare (hidden și creative)și un decodificator de imagini,realizate cu ajutorul claselor Imago și BMP.

1.1.1

1.1.2 Obiective

După cum a fost menționat anterior,obiectivul constă în codificarea/decodificarea mesajului introdus. La momentul redactării acestui document el a fost atins,aplicația fiind capabilă să genereze imagini pe baza unui șablon ales sau doar prin intermediul mesajului.Totuși,aceasta este incompletă,nefiind posibilă prelucrarea a mai mult de 255 de simboluri și folosirea unei imagini deja create drept imagine șablon.

1.1.3 Definiții, Acronime și Abrevieri

Pentru desemnarea claselor și conținutului acestora s-a optat pentru convenția folosită în limbajul Java.Alegerea numelui „ Imago” pentru a reprezenta clasa imagine a fost făcută pe baza unei practici personale.

ASCII:American Standard Code for Information Interchange,standard de codificare a simbolurilor pe un octet;acesta este folosit pentru memorarea mesajului

BMP:BitMap,standard de codificare a imaginilor independent de sistemul de calcul folosit;în cadrul aplicației este folosită varianta cu 24 biți

1.1.4 Tehnologiile utilizate

Pentru scrierea,editarea,compilarea și depanarea codului sursă s-a folosit mediul de programare C++ Builder 6.

Pentru citirea fișierelor în format hexazecimal a fost folosit HxD(https://mh-nexus.de/en/hxd/).

Pentru tehnoredactarea documentației a fost folosit Google Docs.

1.2 Cerințe specifice

Versiunea curentă a aplicației dispune de următoarele funcționalități:

- introducerea unui șir de simboluri
- codificare pe baza unui șablon ales sau din text simplu
- vizualizarea imaginilor de intrare(șablon) și ieșire(imaginea generată)

2 Funcționalitate

2.1 Descriere

Formatul BMP folosit pentru salvarea imaginilor presupune stocarea octeților prin două anteturi și un tablou de pixeli.Cel dintâi antet memorează dimensiunea întregului fișier și adresa de la care începe tabloul de pixeli.Următorul antet(DIB-Device Independent Bitmap) salvează detalii tehnice precum dimensiunile imaginii,numărul de biți per pixel ș.a.m.d. Pe lângă scrierea pixelilor folosiți se mai adaugă și niște octeți de padding pentru a facilita citirea informației de către microprocesor(principiu similar regăsit și în memorarea obiectelor de tipul unei clase)

Pentru a prelucra fișierele și mesajele au fost create două clase,Imago și BMP. Cea dintâi clasă are rolul de a memora datele despre imaginea codificată(textul de inserat,anteturile și dimensiunile),iar cea din urmă are rolul de a accesa fișierele de intrare(imaginea șablon) /ieșire și să le prelucreze.

2.1.1 Codificarea

Dat fiind scopul aplicației,codificarea și decodificarea reprezintă cele mai importante facilități implementate. Cele două tipuri de codificări (hidden și creative) se folosesc de formatul prezentat anterior pentru a scrie mesajul introdus la începutul tabloului de pixeli într-o zona contiguă de memorie.

2.1.2 Decodificarea

Pereche a codificării,decodificarea presupune citirea individuală din zona contiguă de memorie a mesajului și transmiterea acestuia către utilizator.

2.2 Fluxul de evenimente

2.2.1 Flux codificare

Utilizatorul având la dispoziția sa două moduri de codificare,la acționarea butonului „Generare imagine” programul verifică starea obiectului checkBoxMode pentru a decide care mod va fi folosit.Ambele moduri salvează inițial textul introdus și determină numele viitoare imagini generate pe baza convenție:nume fișier=image+nr+.bmp,unde nr este calculat pe baza unei variabile statice (counter) ce contorizează numărul de imagini generate.
Implicit este setat modul hidden(metoda createImageHidden()) cu imaginea de intrare „default.bmp” deja aleasă,și presupune copierea primilor 54 de octeți din imaginea șablon spre imaginea generată.Se continuă prin a salva într-un octet dimensiunea mesajului(motivul pentru care acesta nu poate conține mai mult de 255 caractere) și a suprascrie octeții următori cu caracterele ASCII ale mesajului.La final utilizatorul va observa în partea dreaptă a interfeței apariția noii imagini.
La alegerea modului creativ,aplicația apelează metoda createImageCreative().Această metodă va produce mereu o imagine pătratică cu latura de proporțională cu numărul de simboluri ale mesajului.

```
//Fie num=nr simboluri citite
//Vom crea imagini patratice de latura n:
//n=((1+num+k)/2)
//unde,1 reprezinta stocarea nr de simboluri la adresa 36h
//     k=(1+num)%2,pentru a obtine o valoare para
 int n=(1+this->text.Length()+(( 1+this->text.Length() )%2) )/2;
```

calculul lungimii laturii viitoarei imagini

Se continuă prin a determina numărul de octeți de padding necesari prin formula:padding=(4-(n*3)%4)%4,unde n este latura calculată anterior.În rest,stocarea informației se face identic cu prima metodă,având de această dată o imagine mai mică decât cea obținută cu ajutorul șablonului.

2.2.2 Flux decodificare

Indiferent de modul de codificare selectat,decodificarea rămâne identică și presupune căutarea la adresa 36h din fișier a numărului de simboluri memorat și citirea ulterioară a acestora într-un buffer de date. Șirul citit va fi returnat utilizatorului prin aceeași căsuță de introducere a mesajului folosită anterior pentru codificare.

Prin apăsarea butonului „Generare text” este posibilă alegerea imaginii de decodificat.

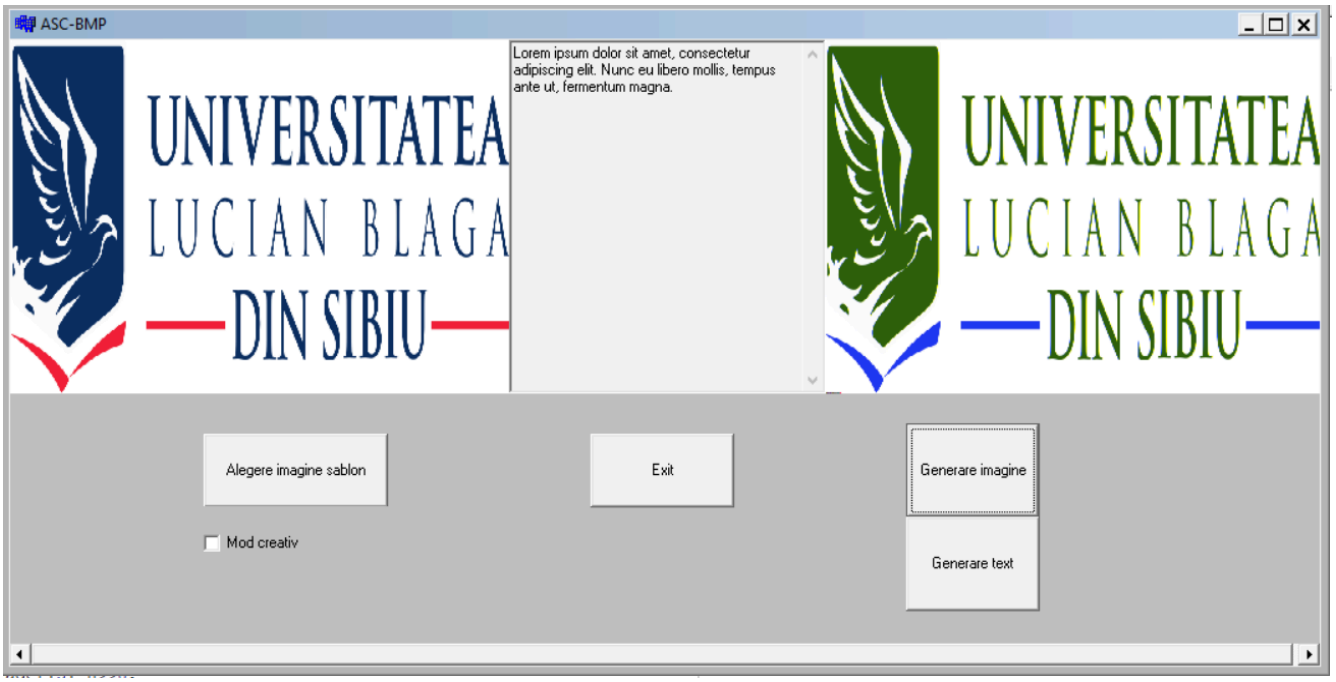
2.2.3 Pre-condiții

Utilizarea aplicației nu impune multe setări preliminare pentru o bună funcționare,fiind deja puse la dispoziție câteva imagini șablon pentru a fi folosite,acestea putând fi înlocuite cu imagini personale(cu condiția ca ele să fie de format bmp).

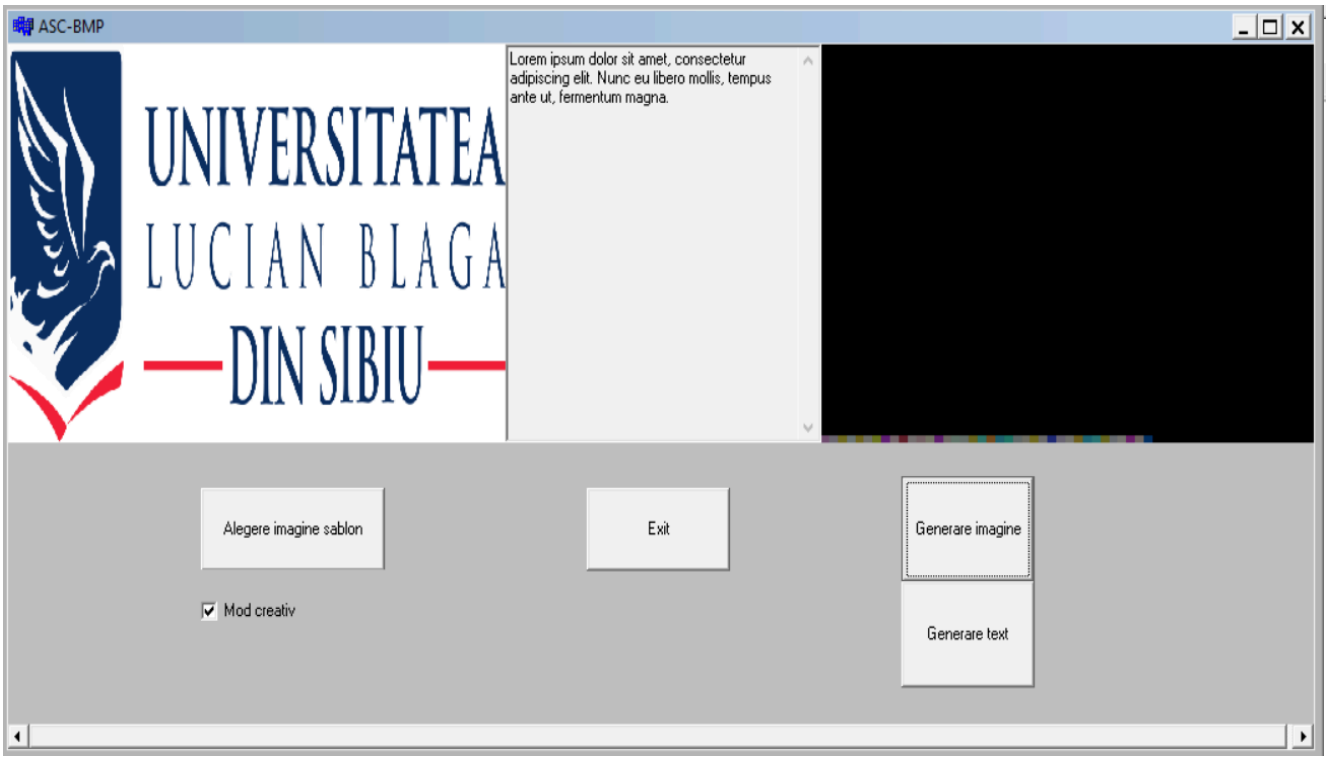
Problema majoră,nerezolvată încă,apare în folosirea unei imagini deja generate drept șablon.S-a observat astfel că dacă în timpul unei rulări o imagine a fost generată și este folosită drept șablon la aceeași rulare,aplicația funcționează corect.Dacă la următoarea rulare se alege aceeași imagine,utilizatorul va fi anunțat că nu a ales un fișier valid.Cauza este ștergerea conținutului la a doua rulare.

2.2.4 Post-condiții

Utilizatorul va observa de o parte și alta a căsuței de scriere două spații goale.După alegerea șablonului și codificarea acestuia,două imagini noi vor fi generate pentru a compara rezultatele.Rezultatul codificării va fi mereu salvat în directorul „imagini_generate” în cazul în care se dorește preluarea lor.



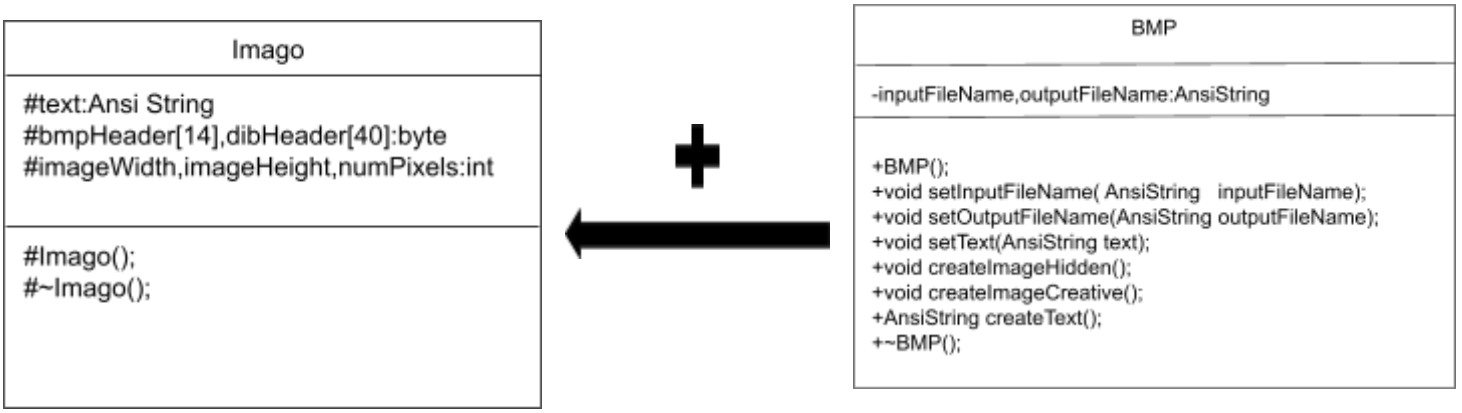
modul hidden



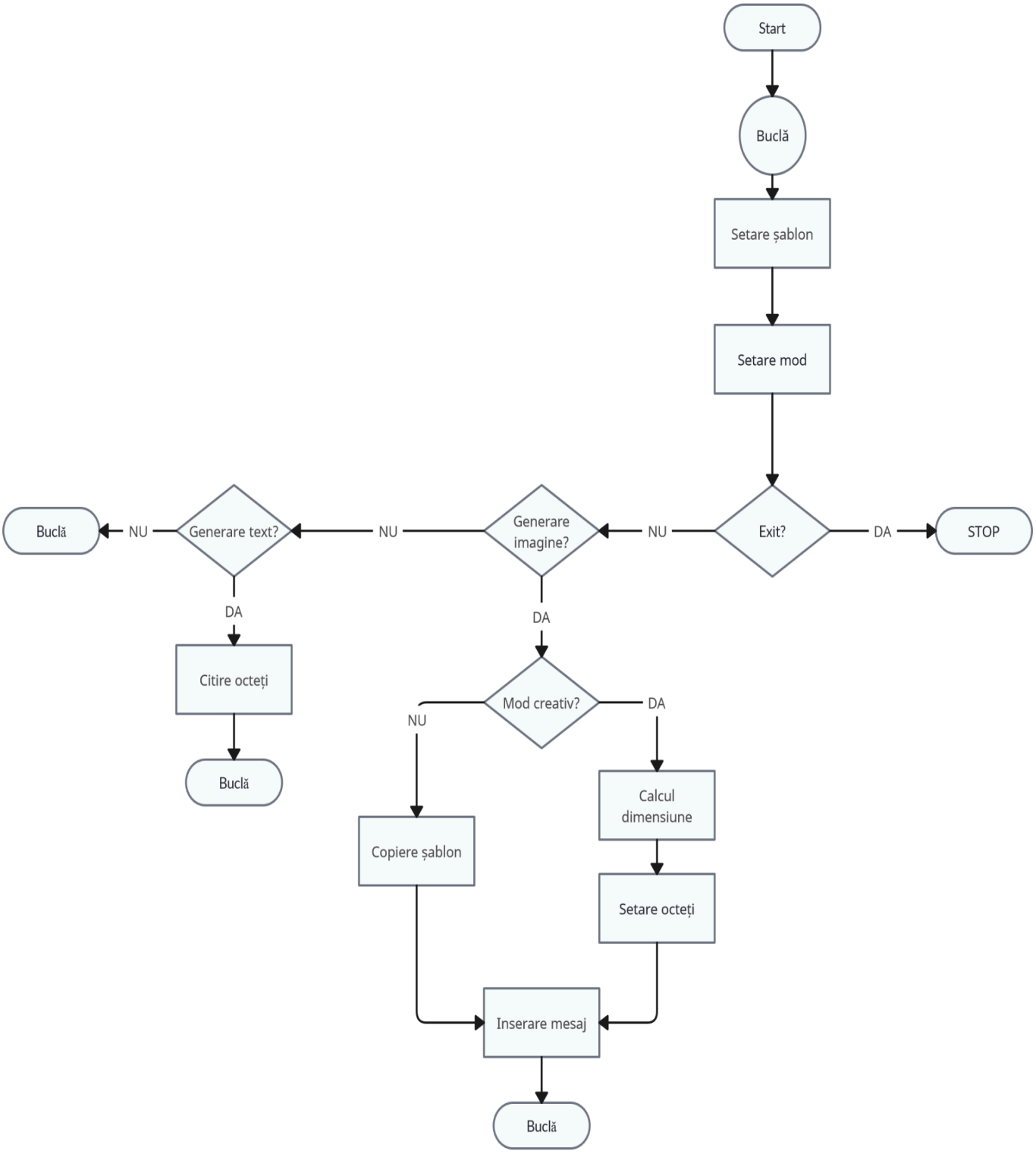
modul creativ

3 Implementare

3.1 Diagrama de clase



3.2 Descriere detaliată



4 Bibliografie

[1]Macarie Breazu, (2002), Programare Orientată pe Obiecte.Principii,Editura Universității "Lucian Blaga" din Sibiu, Sibiu

[2] BMP file format -https://en.wikipedia.org/wiki/BMP_file_format

[3]cplusplus fstream -https://cplusplus.com/reference/fstream/fstream/

[4] Embarcadero C++ Builder- http://www.functionx.com/cppbuilder/

[5] VCL -RAD Studio - http://docwiki.embarcadero.com/RADStudio/Tokyo/en/VCL