Modul de procesare al imaginii

**Autori**

Cobuz Cezara

Colibaba Vali

Focșa Marian

Groza Vasile

Miron Andrei

Țifui Vali

**Grupa**

B5

**An**

2

Cuprins

[1. Descriere 3](#_Toc448778751)

[2. Domenii 3](#_Toc448778752)

[3. Acționari / Interese 3](#_Toc448778753)

[Firma de programatori 3](#_Toc448778754)

[Firma arhitecturală 3](#_Toc448778755)

[4. Actori / Obiective 4](#_Toc448778756)

[Utilizator uman 4](#_Toc448778757)

[API-uri 4](#_Toc448778758)

[5. Scenarii de utilizare 4](#_Toc448778759)

[5.1. Utilizatorul incarca o imagine 4](#_Toc448778760)

[5.1.1 Obiective 4](#_Toc448778761)

[5.1.2. Scenariu 4](#_Toc448778762)

[5.1.3. Extensii 5](#_Toc448778763)

[5.2 Procesarea imaginii 5](#_Toc448778764)

[5.2.1 Obiective 5](#_Toc448778765)

[5.2.2 Scenariu 5](#_Toc448778766)

[5.2.3. Extensii 6](#_Toc448778767)

[5.3 Procesarea imaginii via API 6](#_Toc448778768)

[5.3.1 Obiective 6](#_Toc448778769)

[5.3.2. Scenariu 6](#_Toc448778770)

[5.3.3. Extensii 7](#_Toc448778771)

# Descriere

Modulul de prelucrare al imaginii are ca scop, pe scurt, citirea și prelucrarea unui *blueprint arhitectural al unui etaj (cu extensibilitate pentru mai multe etaje)*, oferind modului de interpretare o colecție de date, stocate în formatul propus de către modulul ce se ocupă cu modelul.

Acesta va oferi o interfață grafică care pune la dispoziție utilizatorului posibilitatea de a interacționa cu aplicația – în principiu încărcarea unei imagini de pe propriul computer, stocată într-un anumit format, apoi eventual prelucrarea acesteia în formatul potrivit aplicației, dacă aceasta este într-un format diferit; De asemenea se pune la dispoziție afișarea succintă a unei descrieri despre imagine și posibilitatea pornirii prelucrării imaginii, cu afișarea informațiilor despre progres pe parcursul operațiilor.

Modulul pune la dispoziție și un mini API, o interfață prin care modulul de interpretare a datelor poate accesa datele procesate.

# Domenii

Se vor descrie scenariile de utilizare atât ale user-ului care încarcă și pornește prelucrarea imaginii, cât și a modulului de interpretare, care cere accesul la date.

# Acționari / Interese

### Firma de programatori

Se ocupă, pentru satisfacerea cerințelor clientului (firma arhitecturală, descrisă mai jos) și pentru satisfacția de a dezvolta o aplicație inovativă, de crearea sistemului de procesare a imaginii și oferirea eventual pe viitor a unui API open-source, care vine însoțit și de un GUI, pentru parsarea blueprint-urilor.

### Firma arhitecturală

Un grup de arhitecți sun reprezentanții firmei. Ei sunt cei care au cerut de la firma de programatori crearea unei aplicații de prelucrare a blueprint-urilor ce identifică un etaj (sau mai multe) al unei clădiri. Necesitatea a apărut datorită timpului imens pe care aceștia îl pierd cu translatarea și interpretarea schițelor de etaje și astfel au nevoie de o aplicație rapidă și sigură care să facă acest lucru pentru ei.

# Actori / Obiective

### Utilizator uman

Este actorul care interacționează direct cu interfața grafică (GUI-ul) modulului de prelucrare a imaginii. Acesta are la dispoziție o serie mică de activități pe care le poate executa, cum ar fi încărcarea în aplicație a unei imagini și pornirea procesării acesteia (cu un progress bar care îi va afișa informații despre statusul curent al progresului).

### API-uri

Modulul de prelucrare a imaginii oferă spre utilizare un API care poate fi accesat de către modulul de interpretare a imaginii (sau de orice alt viitor modul care necesită prelucrarea de astfel de schițe). API-ul primește un request (eventual cu o cale către imagine) și returnează un obiect ce identifică imaginea procesată.

# Scenarii de utilizare

Următoarele cazuri descriu modul prin care utilizatorul uman interacționează cu aplicația.

## 5.1. Utilizatorul încarcă o imagine

### 5.1.1 Obiective

Încărcarea în aplicație a unei imagini de pe propriul PC pentru procesare, eventual si convertire la un format procesabil daca e posibil.

### 5.1.2. Scenariu

1. Utilizatorul deschide interfața grafică pentru modulul curent.
2. Utilizatorul apasă butonul *Load image* după apăsarea căruia va apărea un dialog de deschidere de fișiere (cu filtru per tipurile de imagini acceptate: .**jpg**/**jpeg**, .**png**).
3. Utilizatorul alege imaginea dorită dând *dublu* click pe aceasta sau selectând-o și apăsând butonul *Choose*.
4. Este testată existența și validitatea imaginii.
5. Imaginea este încărcată și reprezentată în memorie (convertită într-un format intern).
6. Utilizatorul a terminat de încărcat imaginea în aplicație și informații despre aceasta îi vor fi afișate în fereastra deschisă (inclusiv imaginea).

### 5.1.3. Extensii

3.Dacă utilizatorul a găsit vreo metodă să suprascrie filtrul aplicației și să nu selecteze o imagine, ci alt tip de fișier, acest lucru va fi testat în timpul validării și i se va afișa un mesaj corespunzător („*Fișierul ales nu este imagine.*”).

4. Dacă imaginea nu este în formatul JPG/JPEG sau PNG și nici nu poate fi interpretată ca o *raster image (generic bitmap, dot matrix data structure)*, un mesaj corespunzător va fi afișat („*Fișierul nu poate fi citit. Format incorect.”).*

## 5.2Procesarea imaginii

### 5.2.1 Obiective

Din imaginea încărcată conform punctului *5.1* se vor extrage informațiile digitale necesare *modulului de interpretare*, folosind reprezentarea datelor în formatul oferit de către *modulul de modelare*.

### 5.2.2 Scenariu

1. Utilizatorul a încărcat deja o imagine în interfața grafică.
2. Utilizatorul apasă butonul de pornire procesare imagine („*Start processing*”).
3. Utilizatorului îi apare un *progress bar* prin care îi este afișat progresul curent al procesării (acesta poate dura ceva timp, în funcție de dimensiunea / calitatea imaginii).
4. Folosind un *procesor grafic* se vor executa următoarele:
   * 1. Se va decupa imaginea în cel mai mic dreptunghi care cuprinde reprezentarea imaginii, având o dimensiune *WxH (width x height)*
     2. Se va încerca transpunerea imaginii într-una *clusterizată* în doar două culori – alb și negru, eventual cu transpunerea și în funcție de grosimea liniilor, pentru a ușura identificarea segmentelor și a elimina micile imperfecțiuni (sau alte culori accidentale).
     3. Se va realiza *segmentarea* imaginii și identificarea următoarelor elemente, conform *anexei* atașate documentului:

* pereți
* uși
* ferestre
* scări
* lifturi.
  + 1. Identificarea va consta în specificarea *coordonatelor relative* (în funcție de dimensiunile dreptunghiului calculat la pasul anterior).

1. Utilizatorului îi apare un mesaj corespunzător finalizării procesării imaginii (succes sau eșec)
2. După terminarea procesării, se realizează transpunerea datele procesate pe modelul de reprezentare a datelor oferit de modulul corespunzător.
3. De asemenea, va fi pusă la dispoziție o interfață prin care un modul extern (*ex. cel de interpretare*) poate prelua modelul curent reprezentat și executa diverse operații pe el (*ex. interpretarea acestuia*).

### 5.2.3. Extensii

1. Dacă nu a încărcat o imagine, nu va putea apăsa butonul ce pornește procesarea imaginii; va trebui să execute use-case-ul 5.1 întâi.

5. Dacă procesarea imaginii eșuează (din diverse motive), utilizatorului îi apare un mesaj de eroare și este rugat să încerce din nou procesarea imaginii (sau a altei imagini).

Următorul caz descrie interacțiunea unei aplicații externe (*sau interne, modulul de interpretare*) cu API-ul oferit de modul.

## 5.3Procesarea imaginii via API

### 5.3.1 Obiective

Se urmărește extragerea de informații prelucrate despre o imagine din acest modul, de către un alt modul extern. Acest lucru se realizează prin interfața / API-ul oferit. Spre exterior pot fi oferite informații despre ultima imagine procesată (procesarea pornită din interfața grafică) sau se poate cere procesarea unei noi imagini, prin specificarea *căii* acesteia.

### 5.3.2. Scenariu

1. Aplicația / modulul extern apelează API-ul, poate oferi sau nu ca parametru calea către o imagine.
   1. Dacă nu oferă calea către o imagine.
      1. Modulul curent verifică dacă a existat o imagine care să fie procesată conform 5.2.
      2. Dacă a existat, îi va returna aplicației externe modelul de imagine procesată pentru a o folosi / interpreta
   2. Dacă oferă calea către o imagine.
      1. Modulul curent verifică existența imaginii
      2. Modulul curent verifică validitatea imaginii
      3. Modulul curent începe parsarea imaginii primite conform 5.2. În acest timp, aplicația care a făcut requestul este pusă în așteptare.
      4. La terminarea procesării, modulul curent returnează modelul de imagine procesată pe care aplicația îl așteaptă.
2. Aplicația apelantă folosește / interpretează modelul de date primit..

### 5.3.3. Extensii

1.1.1. Dacă nu a existat o imagine care să fie procesată, îi va returna un status negativ aplicației apelante.

1.2.1/1.2.2 Dacă imaginea este inexistentă sau invalidă, îi va returna un status negativ aplicației apelante.

1. În cazul unei erori de comunicare, modulul curent va loga erorile (ex. va afișa un mesaj sau le va scrie într-un fișier text)