

Universitatea Politehnica Bucuresti
Facultatea de Automatica si Calculatoare



Preprocessing Graphical User Interface

Documentul de Proiectare a Solutiei Aplicatiei Software
(Software Design Document)

Echipa WiseGUIs:

Pavel Jimmy
Luchian Liliana
Zamfir Mihai
Sevan Roxana
Neculai Gabriel
Popescu Bogdan Ionut
Pasca Luiza

Cuprins

1. Scopul documentului

2. Conținutul documentului

3. Modelul datelor

- 3.1 Structuri de date globale
- 3.2 Structuri de date de legatura
- 3.3 Structuri de date temporare
- 3.4 Formatul fisierelor utilizate

4. Arhitectura sistemului

- 4.1 Diagrama de sistem
- 4.2 Arhitectura sistemului
- 4.3 Descrierea componentelor
- 4.4 Restrictiile de implementare
- 4.5 Interactiunea dintre componente
- 4.6 Tehnologii folosite

5. Modelul interfeței cu utilizatorul

- 5.1 Ferestrele aplicatiei
- 5.2 Succesiunea interfetelor

6. Elemente de testare

- 6.1 Componente critice
- 6.2 Alternative
- 6.3 Scenarii de test

1. Scopul documentului

Acest document are rolul de a descrie soluția proiectată pentru sistemul software “Preprocessing Graphical User Interface”.

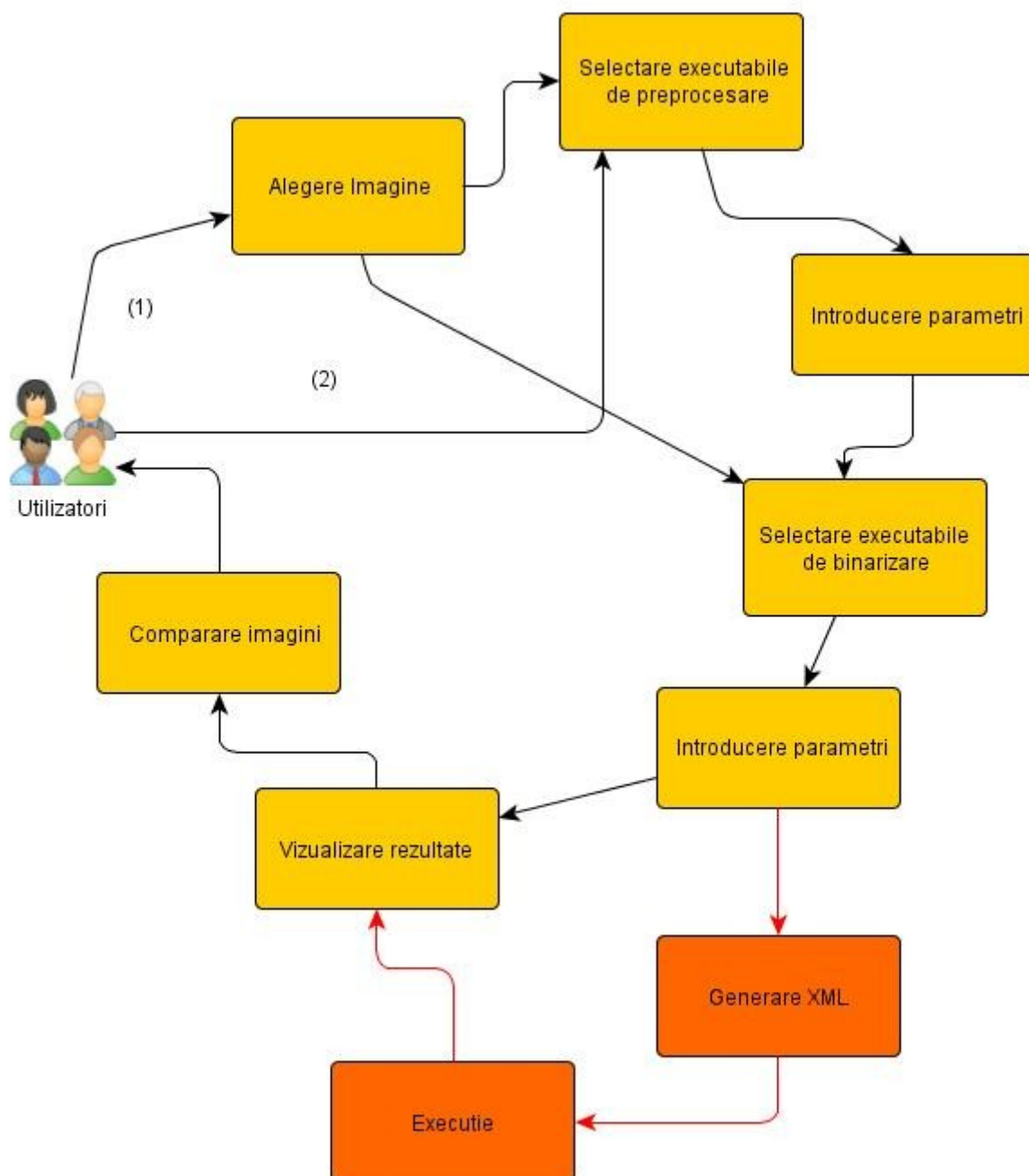
2. Conținutul documentului

Documentul este format din patru parti:

- 1) **Modelul datelor (Data Design)** - prezintă structurile de date folosite în cadrul soluției.
- 2) **Modelul arhitectural și modelul componentelor (Architectural Design)** - prezintă arhitectura sistemului sub o structură ierarhică de elemente interconectate.
- 3) **Modelul interfeței cu utilizatorul (User Interface Design)** - prezintă interfața cu utilizatorul și succesiunea ferestrelor acestuia
- 4) **Elemente de testare (Testing Issues)** - prezintă componentele critice și alternative de proiectare a acestora.

3. Arhitectura sistemului

3.1 Diagrama de sistem



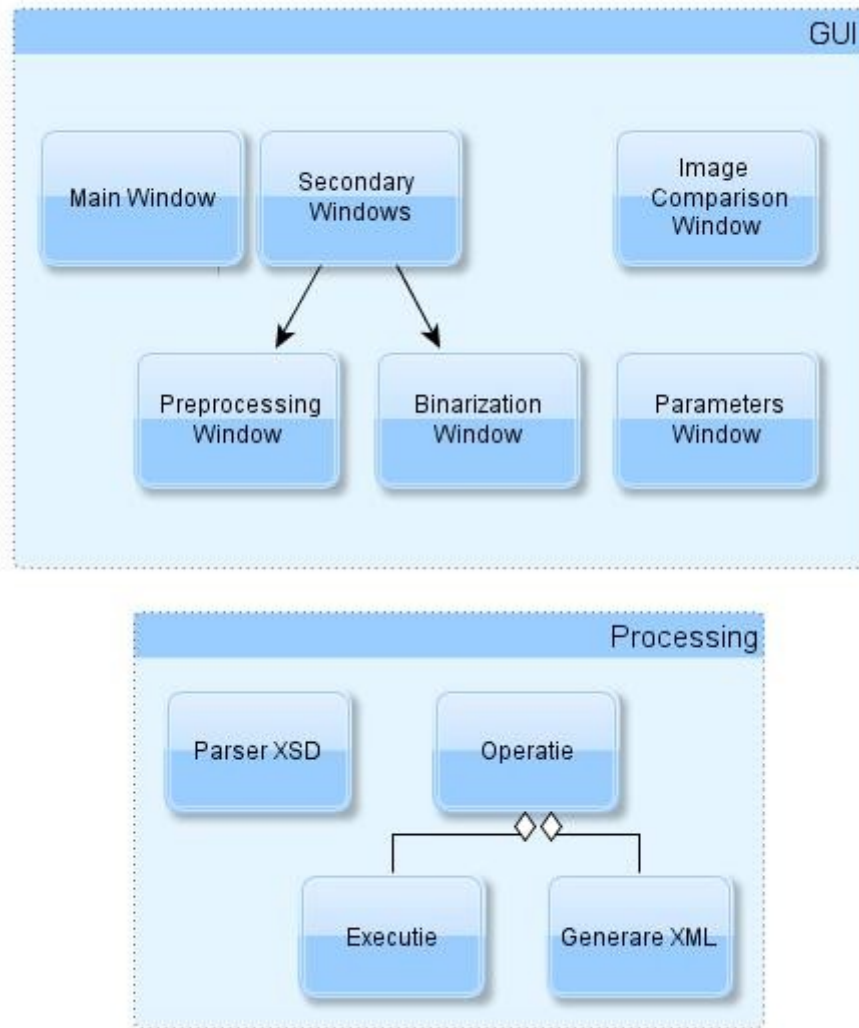
Din **perspectiva utilizatorului**, workflow-ul este urmatorul:

- deschide aplicatia
- alege o cale catre un fisier imagine si il vizualizeaza(1)
- (optional) alege unul sau mai multe executabile de preprocesare prin care prelucreaza imaginea originara; introduce parametrii specifici acestora
 - vizualizeaza rezultatul direct pe imaginea originara
 - alege unul sau mai multe executabile de binarizare; introduce parametrii specifici acestora
 - vizualizeaza rezultatele - cate o imagine distincta pentru fiecare executie
 - (optional) alege sa compare 2 imagini, prin suprapunere
 - user-ul poate relua procesul de prelucrare, pe aceeasi imagine (2)

Din **perspectiva aplicatiei** se adauga urmatoarele operatii la workflow-ul descris mai sus, intre introducerea parametrilor si vizualizarea rezultatelor:

- dupa introducerea parametrilor **se genereaza un XML specific** executabilului, care contine valorile acestor parametri
 - **se executa toate programele** de preprocesare/de binarizare selectate de user la un moment dat, folosind XML-ul generat
 - rezultatele sunt afisate utilizatorului

3.2 Arhitectura sistemului



3.3 Descrierea componentelor

1) Modulul GUI

Permite interactiunea cu utilizatorul, setarea parametrilor pentru executabilele de prelucrare a imaginilor.

Submodule:

- ➔ **Main Window** - afiseaza imaginea originala si imaginea rezultat; este, de asemenea, modulul principal al aplicatiei, modulul de control;
- ➔ **Secondary Windows** - modul din care vor fi derivate doua ferestre (Preprocessing Window si Binarization Window);
- ➔ **Parameters Window** - modul din care vor fi derivate ferestre cu parametri specifici fiecarui executabil
- ➔ **Image Comparison Window** - se ocupa de compararea a doua imagini binare; le va afisa suprapuse, astfel incat utilizatorul sa poata observa diferentele

2) Modulul Processing

Efectueaza prelucrarile din backend.

Submodule:

➔ **Parser XSD** - parseaza xsd-ul specific operatiei alese de utilizator oferind informatii esentiale pentru generarea xml-ului.

➔ **Operatie** - reprezinta o abstractizare a unui executabil; incapsuleaza valorile parametrilor.

➔ **Generare XML** - folosit de *Operatie* pentru a genera xml-ul necesar rularii executabilului.

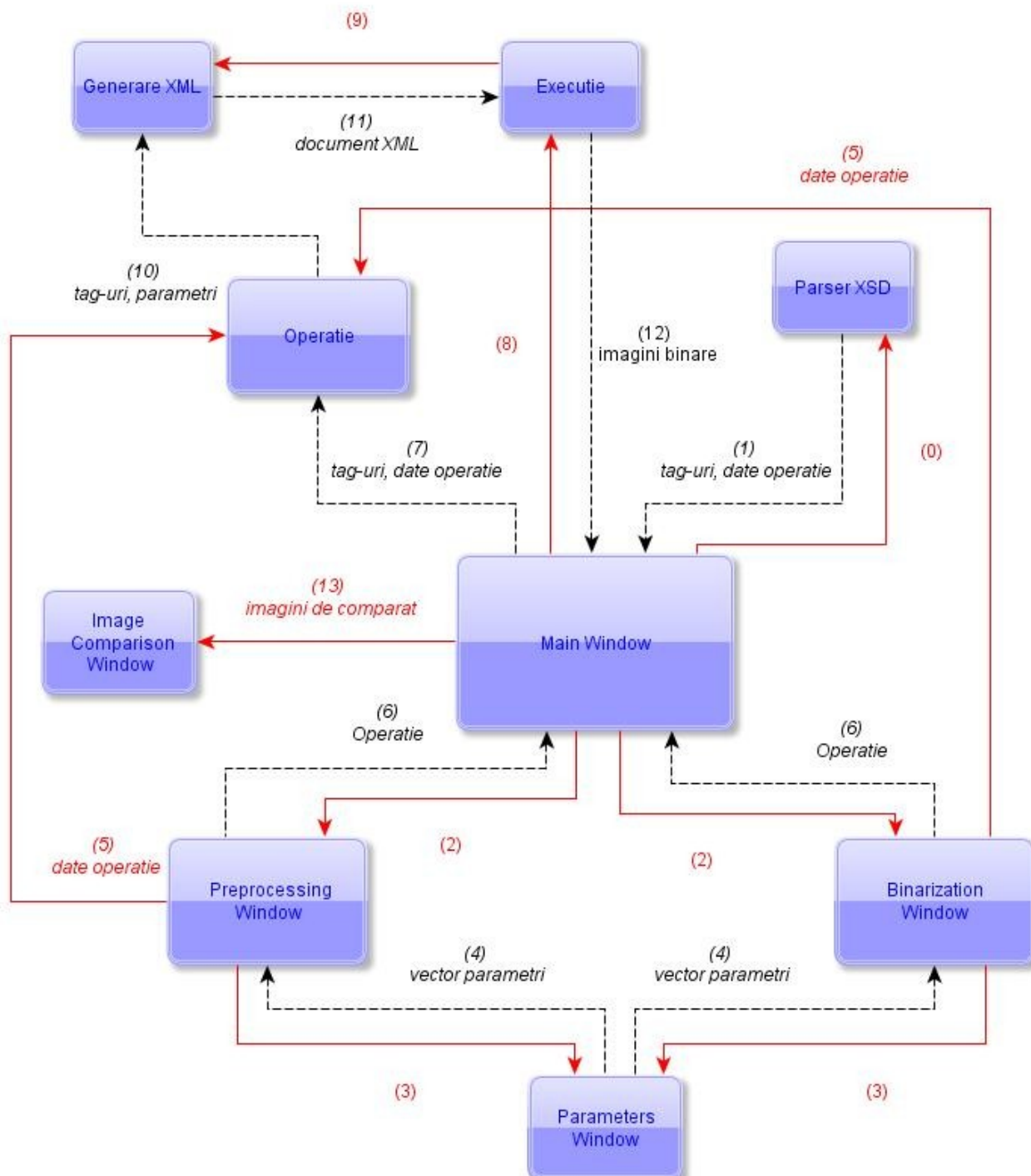
➔ **Executie** - folosit de *Operatie* pentru a rula executabilul, utilizand xml-ul generat mai sus.

3.4 Restrictiile de implementare

Utilizatorul este limitat la 5 operatii de binarizare in acelasi timp si la maxim 10 imagini afisate in panelul din dreapta din fereastra principala.

Numele xsd-ului va fi acelasi cu numele operatiei si al executabilului (ex: operatia „rotate” va parsa un fisier xsd cu numele „rotate.xsd” si va rula executabilul cu numele „rotate.exe”).

3.5 Interactiunea dintre componente



(0) - modulul "Main Window" apeleaza modulul "Parser XSD" (se parseaza o singura data xsd-urile, la inceput; informatia astfel obtinuta - tag-urile - vor fi folosite si refolosite ulterior)

(1) - modulul "Parser XSD" intoarce o structura de date ce va contine tag-urile din fiecare xsd, specifice unui anumit tip de executabil

(2) - modulul "Main Window" apeleaza "Preprocessing Window" sau "Binarization Window" (fie in ordinea Preprocessing - Binarization, fie doar Binarization).

(3) - modulele "Preprocessing Window" sau "Binarization Window" apeleaza "Parameters Window"

(4) - "Parameters Window" intoarce un vector cu valorile parametrilor specifici unui executabil

(5) - "Preprocessing Window" sau "Binarization Window" apeleaza modulul "Operatie", pentru fiecare executabil selectat de catre utilizator; modulului "Operatie" i se transmit valorile parametrilor pentru acea operatie

(6) - "Preprocessing Window" sau "Binarization Window" intorc un vector de instante "Operatie" in "Main Window"

(7) - "Main Window" completeaza fiecare instanta "Operatie" cu alte date (de ex., numele fisierului de intrare)

(8) - "Main Window" apeleaza submodulul "Executie" din modulul "Operatie"

(9) - submodulul "Executie" apeleaza submodulul "Generare XML" din modulul "Operatie"

(10) - submodulul "Generare XML" preia date din modulul "Operatie", necesare generarii XML-ului

(11) - submodulul "Generare XML" intoarce submodulului "Executie" numele documentului XML necesar executiei programului

(12) - submodulul "Executie" intoarce numele imaginii binare rezultate in urma executiei programului

(13) - "Main Window" apeleaza modulul "Image Comparison Window", care va prelua 2 imagini binare si le va afisa suprapuse

3.6 Tehnologii folosite

Pentru implementare se vor folosi limbajul Java si framework-ul Java Swing.

4. Modelul datelor

4.1 Structuri de date globale

Nu exista structuri de date globale.

4.2 Structuri de date de legatura

- Calea fisierului de intrare intre Main Window si Secondary Windows;
- Tag-urile obtinute din xsd-uri, necesare generarii xml-urilor intre „Main Window” si „Generare XML”
 - Vector cu parametri intre „Parameters Window” si „Secondary Windows”;
 - Va exista un vector de structuri de date specifice fiecarui executabil, care se va trimite de la „Secondary Windows ” la „Main Window”. Structura de date va retine:
 - ➔ numele executabilului
 - ➔ numarul parametrilor
 - ➔ un vector de string-uri care contin valorile parametrilor alesi de utilizator.

4.3 Structuri de date temporare

Nu se utilizeaza structuri de date temporare importante.

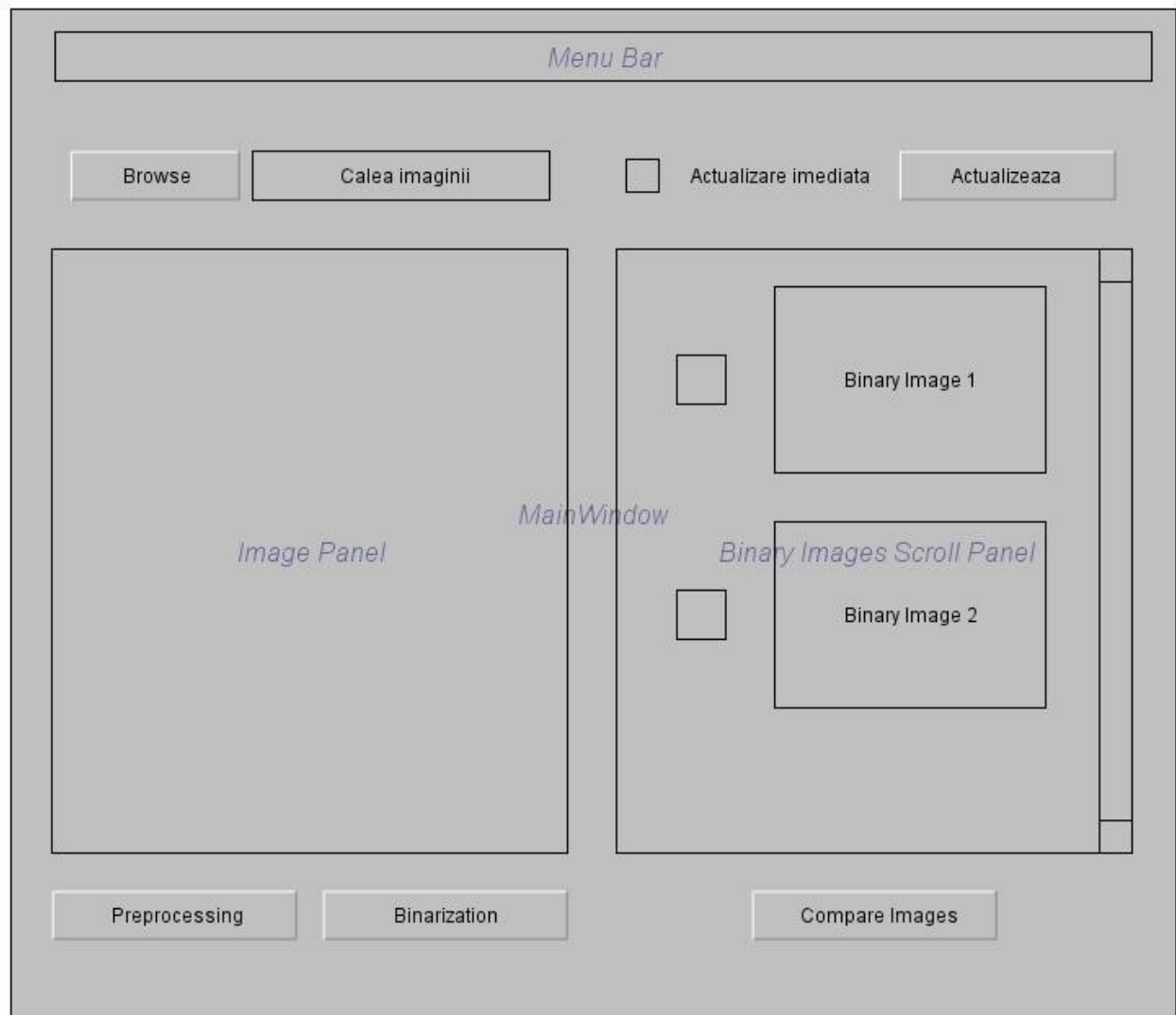
4.4 Formatul fisierelor utilizate

- Imaginile de iesire si de intrare sunt format .jpg;
- Fisiere necesare pentru rularea programelor: tip .xsd si .xml;
- Programele de preprocesare si de binarizare: .exe.

5. Modelul interfetei cu utilizatorul

5.1 Ferestrele aplicatiei

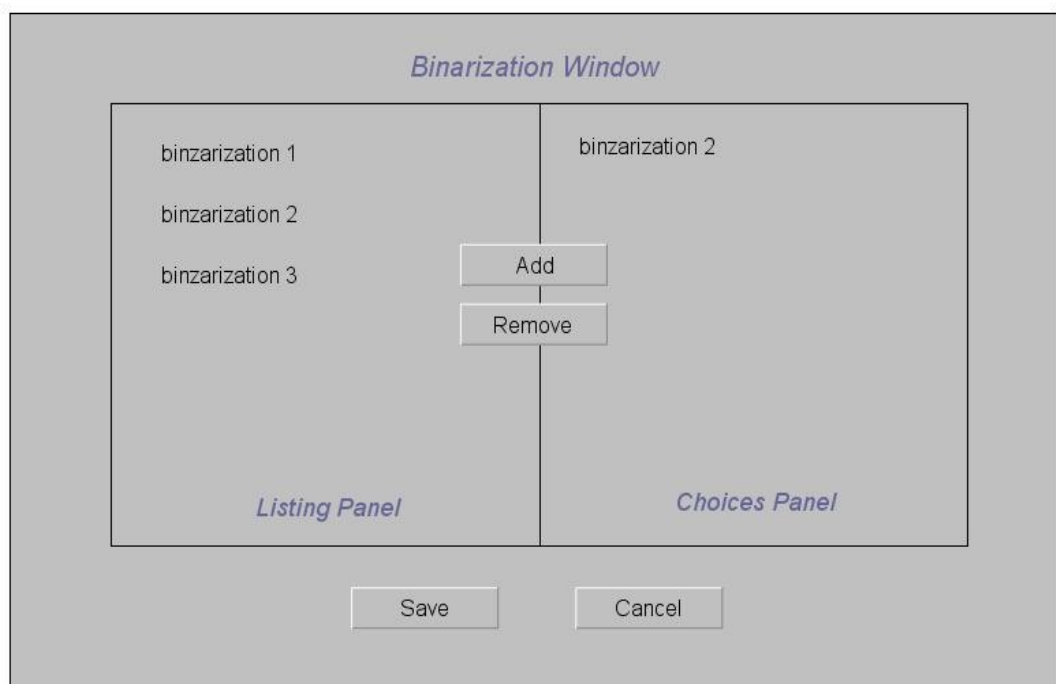
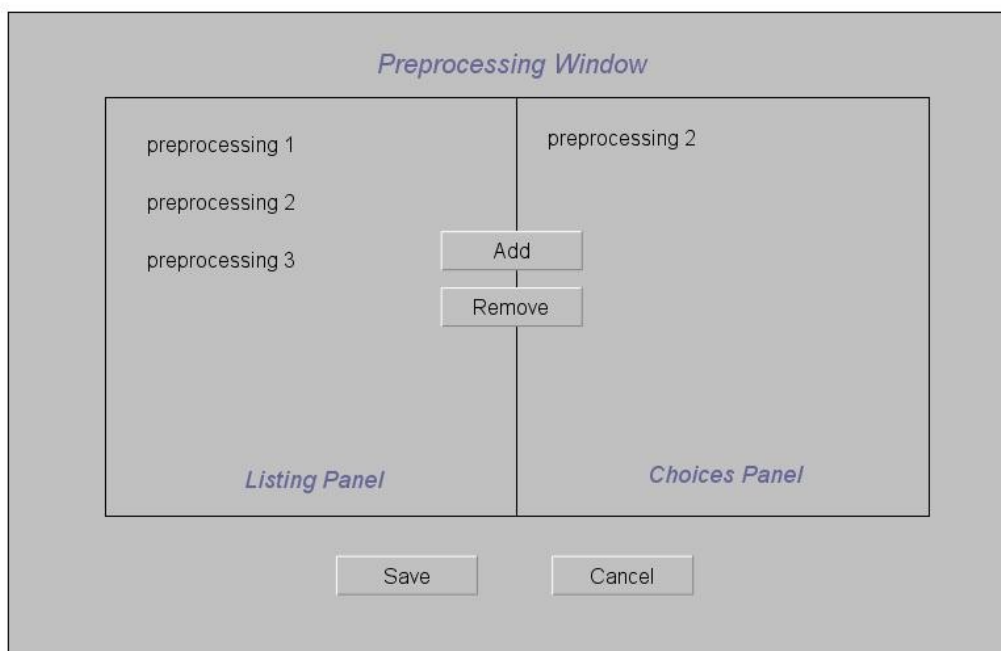
1) Main Window



- butonul „Browse” permite selectarea caili fisierului de intrare, cale ce se va afisa intr-un textbox in partea dreapta a acestui buton
- butoanele “Processing” si “Binarization” deschid ferestrele “Processing Window”, respectiv “Binarization Window”
- butonul “Compare Images” va deschide fereastra “Images Comparison Window”; va fi *disabled* cat timp nu s-au selectat fix doua imagini pentru comparare
- checkbox-ul “Actualizare imediata” - la bifarea acestuia, imaginile binare se vor actualiza automat, imediat dupa ce utilizatorul selecteaza executabilele de binarizare / preprocesare

- butonul “Actualizeaza” va actualiza imaginile binare; va fi *disabled* cat timp este bifat checkbox-ul specificat mai sus
- panel-ul din dreapta va afisa imaginea initiala
- panel-ul din stanga va afisa o lista de imagini binare, obtinute in urma rularii executabilelor de binarizare; fiecare imagine va avea in dreptul ei un checkbox - util pentru selectarea imaginilor de comparat, in caz ca se doreste acest lucru; nu se vor putea bifa decat maxim doua imagini.

2) Secondary Windows („Binarization” sau „Preprocessing”)



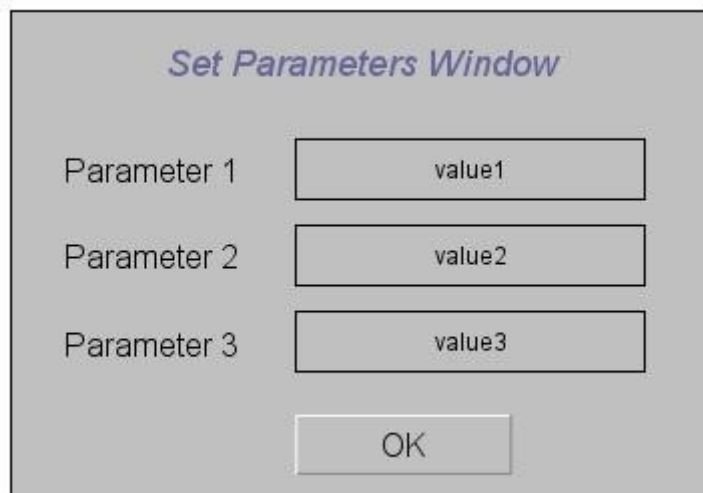
Elemente comune:

- panelul din stanga, in care sunt afisate variantele de executabile
- panelul din dreapta, in care vor fi afisate executabilele selectate de user
- butonul „Add” selecteaza executabilele ce urmeaza a fi aplicate si deschide fereastra “Parameters Window”
- butonul „Remove” elimina din lista din dreapta un executabil selectat
- butonul „Save” intoarce la „Main Window”, salvand lista de operatii ce trebuie executate;
- butonul „Cancel” intoarce la „Main Window” fara niciun efect.

Diferente:

- la “Preprocessing” - executabilele selectate sunt eliminate din lista din stanga - un executabil nu poate fi rulat decat o singura data intr-un ciclu, cu orice configuratie valabila de parametri
- la “Binarization” - executabilele selectate nu sunt eliminate din lista din stanga; se permite rularea aceluiasi executabil de mai multe ori intr-un ciclu, cu diferite configuratii de parametri

3) Parameters Window



The image shows a dialog box titled "Set Parameters Window". It contains three labeled text input fields: "Parameter 1" with the value "value1", "Parameter 2" with the value "value2", and "Parameter 3" with the value "value3". Below these fields is an "OK" button.

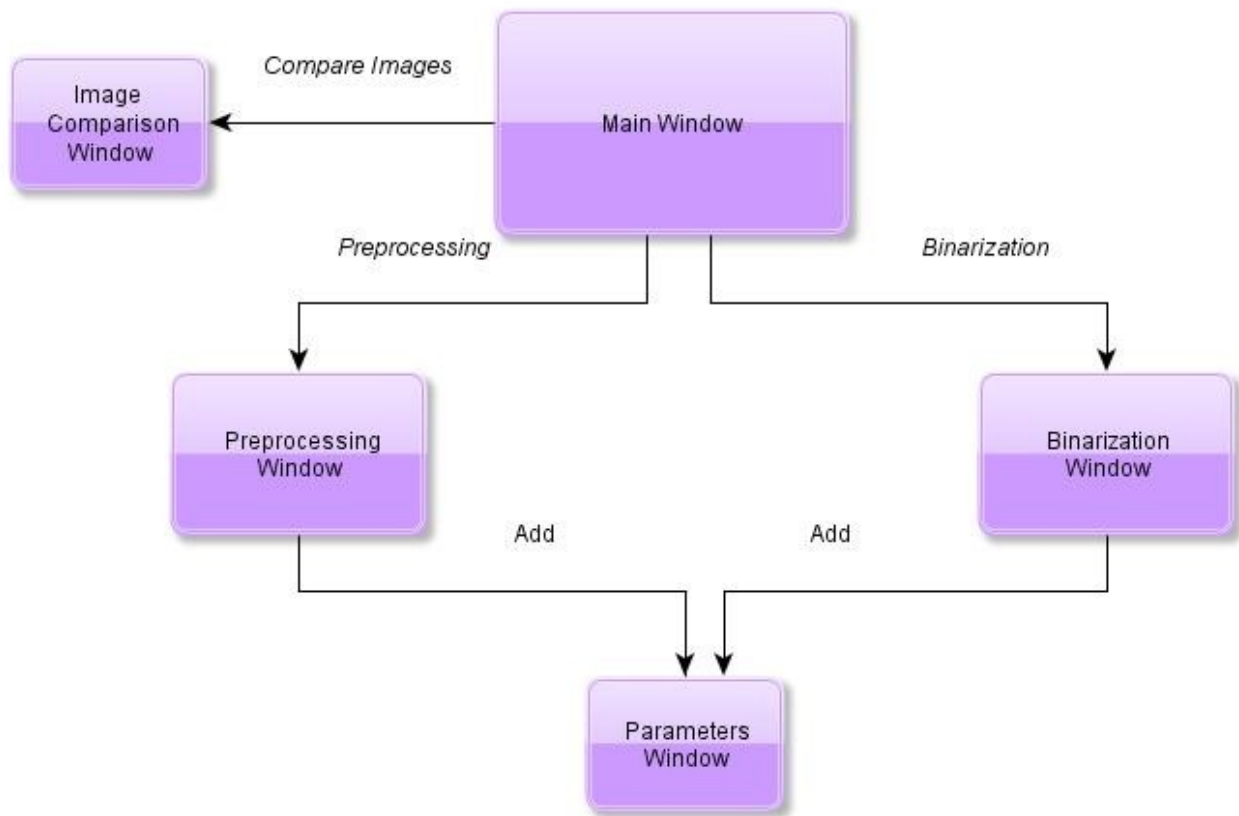
Contine:

- un numar de textbox-uri unde vor fi introduse valorile parametrilor, numar dat de tipul executabilului ales
- butonul „OK” care salveaza parametrii si intoarce la „Secondary Windows”

4) Images Comparison Window

Aceasta fereastra afiseaza doua imagini binare suprapuse. Datorita simplitatii design-ului, nu am mai ilustrat-o grafic in acest document.

5.2 Succesiunea interfetelor



Din "Main Window" se poate trece in "Preprocessing Window", prin apasarea butonului „Preprocessing”, respectiv in „Binarization Window”, prin apasarea butonului „Binarization”. Aici se pot selecta operatiile ce vor fi efectuate asupra imaginii de intrare. Prin apasarea butonului „Add” se deschide fereastra „Parameters Window”. La apasarea butonului „OK”, se inchide fereastra si se revine in "Preprocessing / Binarization Window". La apasarea butonului „Save” din aceste ferestre se revine in "Main Window". Tot din „Main Window” se poate trece in „Image Comparison Window”, prin apasarea butonului „Compare Images”.

6. Elemente de testare

6.1 Componente critice

Modulele *Generare XML* si *Executie* - este necesara verificarea generarii corecte a fisierelor XML si a rularii executabilelor.

Sunt importante, de asemenea, preluarea si aplicarea corecta a parametrilor (de exemplu, unghiul rotatiei sa fie specificat corect).

6.2 Alternative

In cazul in care transmiterea datelor intre module nu functioneaza corect, putem implementa cu date globale.

In cazul in care parserul xsd nu va functiona corect, vom programa hard-core.

6.3 Scenarii de test

Testarea de integrare va presupune:

- Testarea schimbului de date intre module;
- Testarea corectitudinii apelurilor intre module.

Testarea functionala va presupune:

- Testarea tuturor butoanelor, interfetelor;
- Testarea comportamentului aplicatiei in cazuri limita (neintroducerea tuturor parametrilor, introducerea incorecta al tipului parametrilor, incalcarea restrictiilor de implementare etc);
- Utilizarea repetata a aplicatiei (de exemplu, executia mai multor seturi de programe de binarizare).
- Spamming-ul butoanelor (de exemplu ce se intampla cand se apasa butonul „Add” de multe ori);
- Testarea tuturor cazurilor posibile de workflow.