Hierarchy Analysis

Modelul datelor

I . Formatele fisierelor:

- .xml
- .xsd
- .java
- .class
- .exe

Fisierele cu extensia .xml sunt obtinute in urma analizei de layout. Au rol de fisiere de intrare/iesire. Se incearca mai intai rularea unui analizator ierarhic pentru fisierele cu aceasta extensie si apoi incarcarea fisierelor de acest tip si transmiterea rezultatului analizei ierarhice.

Fisierele cu extensia .xsd au ca input 2 sectiuni:

-o sectiune de descriere a aplicatiei careia ii este asociat (utilizata pentru sectiunile grafice pentru a determina ce executabile poate rula pentru a-si atinge scopul)

-o sectiune de configurare propriu-zisa a aplicatiei pentru rulare (reprezinta o insiruire a parametrilor posibili pentru aplicatie)

Intrucat aplicatia va fi implementata folosind limbajul de programare java, fisierele sursa vor avea extensia specifica .java. Practic, tot codul proiectului se va afla in fisierele cu acest tip de extensie.

Dupa compilarea fisierelor sursa java, se obtin fisierele cu extensia .class care pot fi citite de java virtual machine si sunt independente de platforma utilizata(Windows, Linux, MAC OS X).

Fisierele cu extensia .exe sunt executabile specifice sistemului de operare Windows. In cadrul proiectului, fisierele .exe vor actiona asupra fisierelor xml de tip layout (care vor

putea fi selectate prin intermediul interfetei grafice), avand ca rezultat producerea fisierului xml, asupra caruia s-a realizat analiza de ierarhie .

II. Structuri de date

1.Structuri de date globale

Structura de date globala folosita este o instanta a clasei *Main* (clasa core a aplicatiei), ce are identificatorul *<todo – name of main function>*. Cu ajutorul acestei structuri de date globale cu rol de intermediar, clasele diferitelor module isi pot accesa reciproc structurile de date interne.

2. Structuri de date de legatura

Pentru realizarea modulului GUI se foloseste structura de date cu rol de legatura: Jpanel. Aceasta ofera functionalitatea crearii interfetei grafice printr-o actiune de tip drag & drop. Fiecarui element grafic (ex. butoane, meniuri etc) ii va fi generat automat formatul functiei pentru diferite actiuni (click stanga, click dreapta, mouse moved, mouse released etc), urmand ca apoi corpul functiei sa fie completat pe parcursul dezvoltarii proiectului folosind cod specific Java.

3. Structuri de date temporare

Nu se utilizeaza structuri de date temporare cu rol important sau presupunand un consum semnificativ de resurse de memorie.

Componente:

1.O clasa care foloseste JAXB-ul pentru a incarca fisierul de configurare. Dupa incarcarea fisierului datele furnizate de acesta sunt folosite pentru a rula executabilul peste fisierele de intrare.

2.O clasa care incarca folosind un parser(tot JAXB-ul) XML-ul rezultat din urma rularii executabilului peste fisierele de intrare si ofera metode de alterare pentru cazul in care este nevoie de corectii.

3.Un UI care imbina primele 2 componente sub o interfata usor de folosit de catre utilizatori.

4.Componenta auxiliara: parserul folosit pentru incarcarea XML-urilor.

Restrictii de implementare a componentelor:

Datorita executabilului de tip .exe (care face analiza propriu zisa) proiectul trebuie lucrat in Windows.

Restrictiile impuse de acelasi executabil prin fisierele de iesire.

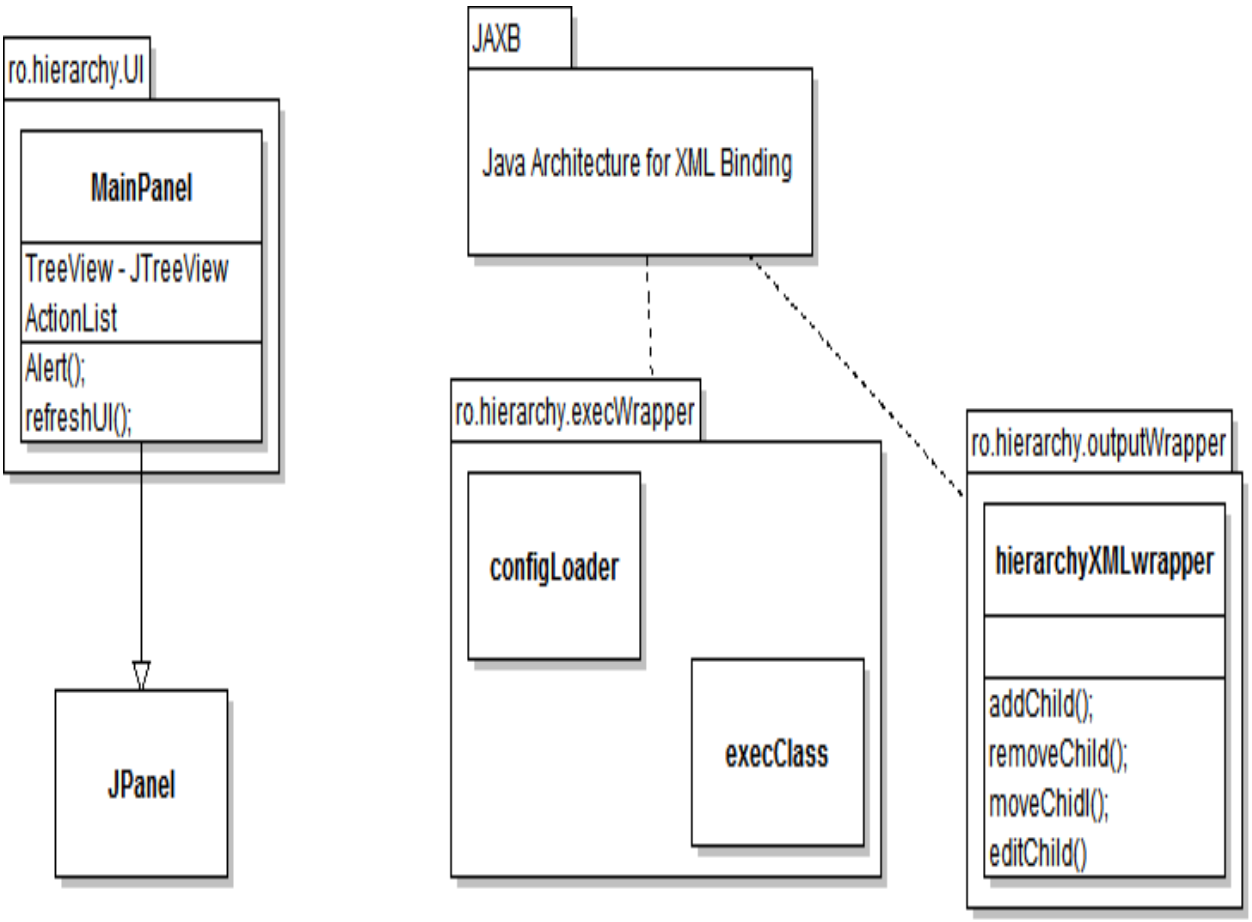
Am ales sa lucram in Java care impune restrictia de masina virtuala la cantitatea de memorie si stack.

Interactiunea dintre componente:

Main-ul face legatura dintre cele 3 componente principale ale sistemului (UI si cele 2 clase)

In UI se vor selecta fisierele de intrare. Apoi folosind prima clasa, acestea sunt trecute prin "black box" (executabilul furnizat) folosindu-se fisierul de configurare.

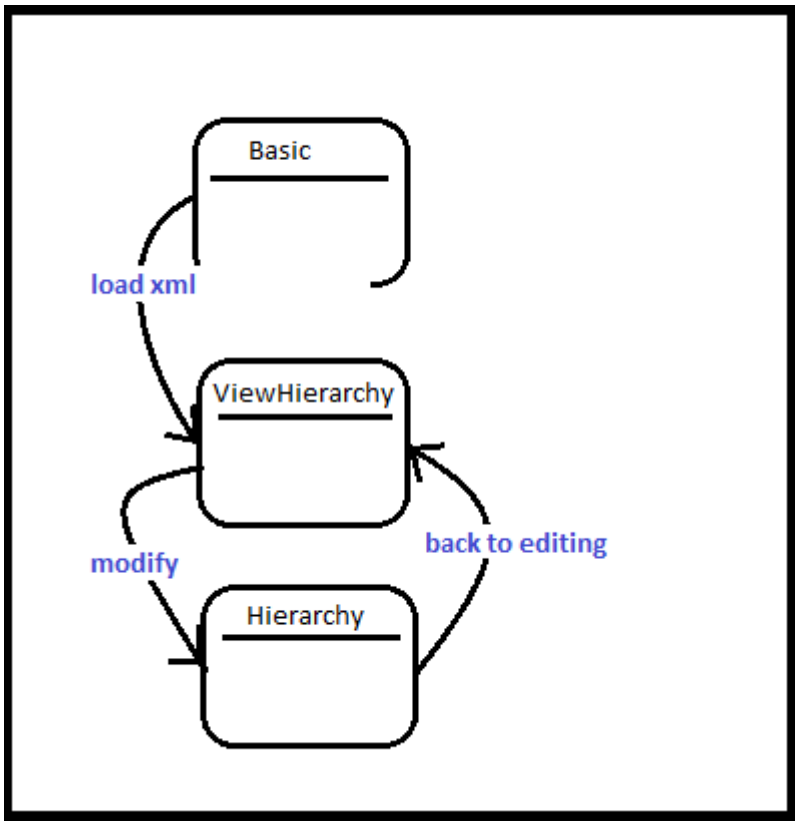
Tot in UI se va putea modifica fisierul de iesire pentru a realiza eventualele corectii.


ro.hierarchy.UI
MainPanel
TreeView - JTreeView
ActionList
Alert();
refreshUI();
JPanel
JAXB
Java Architecture for XML Binding
ro.hierarchy.execWrapper
configLoader
execClass
ro.hierarchy.outputWrapper
hierarchyXMLwrapper
addChild();
removeChild();
moveChidl();
editChild()

3. Modelul interfectei cu utilizatorul

- **Succesiunea interfetelor**

In cadrul modelului interfetei cu utilizatorul, ferestrele aplicatiei sunt afisate potrivit urmatorului flux:

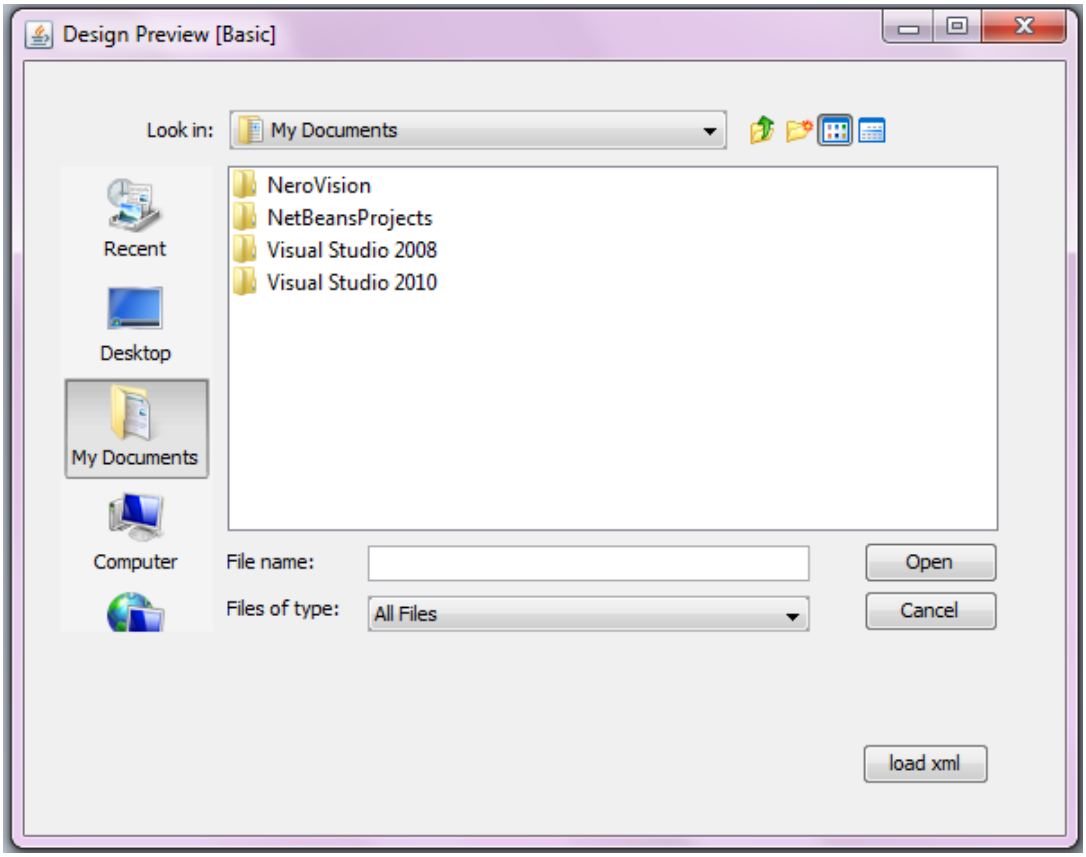


Astfel, utilizarea aplicatiei incepe cu o fererastra de inceput in care utilizatorul selecteaza unul/ mai multe fisiere xml pe care le va concatena intr-o ierarhie cu ajutorul butonului “Generate Hierarchy”.

Apoi, avand ierarhia creata, utilizatorul o va putea modifica prin intermediul unor optiuni existente, iar orice modificare va putea fi revocata.

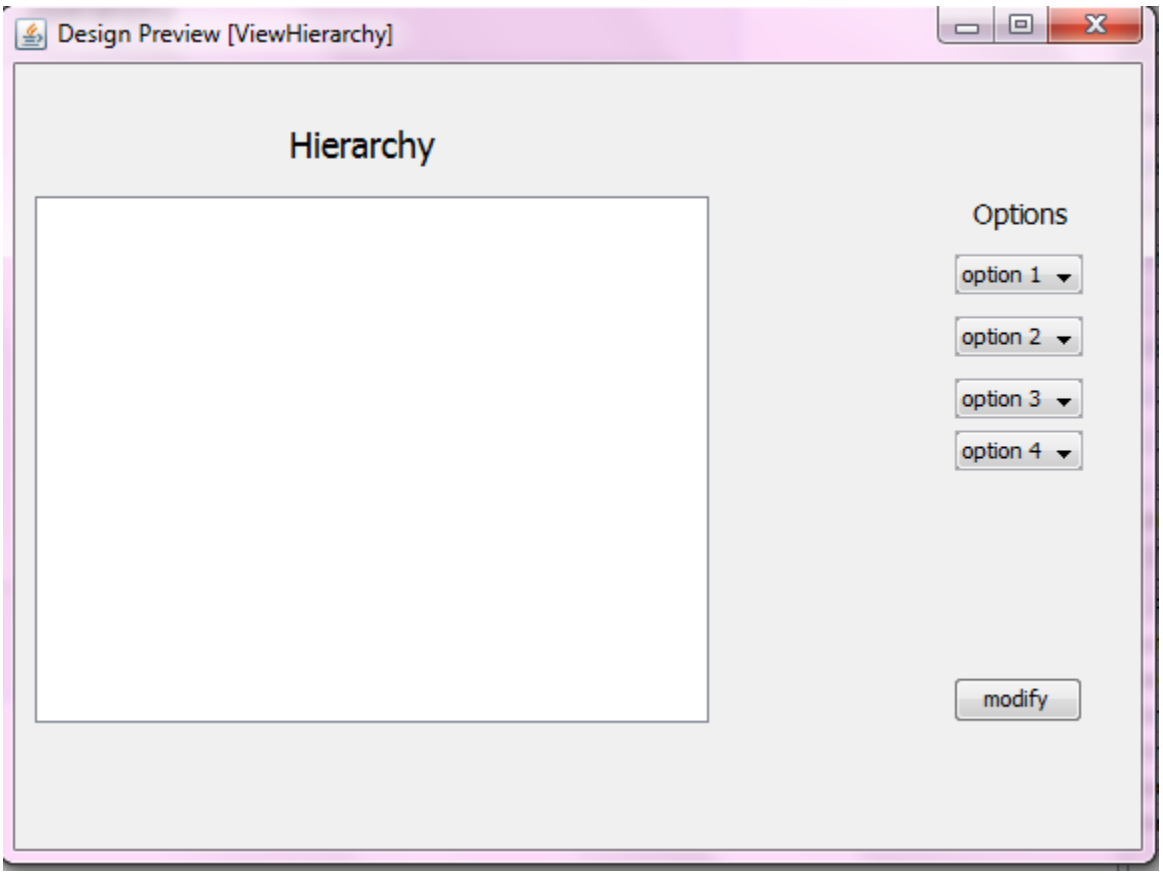
- **Ferestrele aplicatiei**

Fereastra Basic



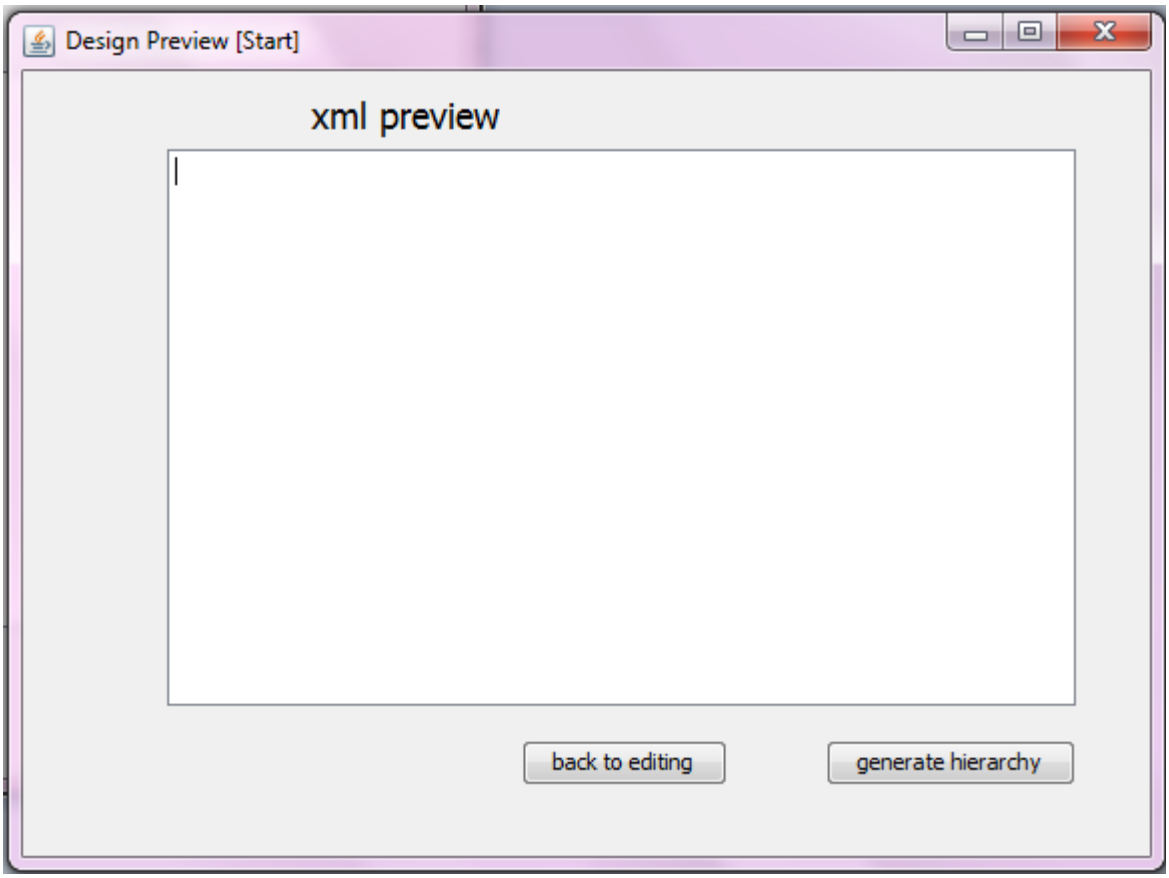
Este fereastra in care utilizatorul incarca fisierele pe care le doreste in ierarhie.

Fereastra ViewHierarchy



Utilizatorul va vizualiza ierarhia si va alege din optiunile disponibile modificari pe care le va aduce acesteia. Cu ajutorul butonului “modify”, modificarea va fi aplicata.

Fereastra Hierarchy



Utilizatorul vede modificarile aduse ierarhiei si poate fie sa salveze modificarile facute (butonul “generate hierarchy”), fie sa continue modificarile (butonul “back to editing”).

4. Elemente de testare

Testarea aplicatiei va consta in a verifica daca functionalitatile sunt indeplinite de fiecare modul in parte si daca toate modulele se conecteaza intr-un tot unitar.

Se iau in considerare atat numarul de fisiere xml cat si rezultatele intoarse de program. Se verifica in cadrul ferestrei “ViewHierarchy” daca ierarhia este corecta, iar in cadrul ferestrei “Hierarchy” se verifica daca modificarile sunt corecte si se poate salva fisierul.