**Documentul de Proiectare a Solutiei Aplicatiei Software**

**(Software Design Document)**

Versiunea 1.0

22 Octombrie, 2012

**Document Processing Manager**

**Echipa The Furry-Footed Gnomes,**

**Facultatea de Automatica si Calculatoare, Universitatea Politehnica, Bucuresti**

Cuprins

1. Scopul documentului

2. Continutul documentului

3. Modelul datelor

3.1. Structuri de date globale

3.2. Structuri de date de legătură

3.3. Structuri de date temporare

3.4. Formatul fişierelor utilizate

4. Modelul arhitectural şi modelul componentelor

4.1. Diagrama de arhitectură

4.2. Descrierea componentelor

4.3. Restricţiile de implementare

4.4. Interacţiunea dintre componente

5. Modelul interfeţei cu utilizatorul

5.1. Succesiunea interfeţelor

5.2. Ferestrele aplicaţiei

6. Elemente de testare

6.1. Componente critice

6.2. Alternative

1. Scopul documentului

Acest document are rolul de a descrie acurat si complet solutia proiectata pentru sistemul software*Document Processing Manager.* Documentul serveste drept ghid unic de construire a solutiei pentru echipa de dezvoltare a proiectului.

2. Continutul documentului

Documentul este format din patru sectiuni esentiale :

* **Modelul datelo**r prezinta structura aplicatiei din punct de vedere al modulelor
* **Modelul arhitectural si modelul componentelor** prezinta sabloanele si compenentele arhitecturale folosite si modul de implementare a interactiunii dintre ele.
* **Modelul interfetei cu utilizatorul** prezinta succesiunea interfetelor si design-ul fiecareia
* **Elementele de testare** prezinta componentele critice si alternative de proiectare a acestora

3. Modelul datelor

**3.1. Structuri de date globale**

Toate executabilele sistemului OCR se vor afla într-un folder, schemele fișierelor XML de intrare se vor afla într-un folder(distinct fata de folderul in care se afla executabilele), schemele ce prezintă formatele de ieșire impuse se vor afla într-un folder(distinct de celelalte 2 foldere).

**3.2. Structuri de date de legatura**

Fisierele XML definite si calculate in procesul aplicatiei.Aceste fisiere sunt folosite pentru cominicarea interna intre module.Lansarea in executie a unui executabil se face pe baza XML-ului care contine toti parametrii inainte de lansare.

**3.3. Structuri de date temporare**

Pe langa principalele structuri de date, exista si alte obiecte a caror durata de viata fie se limiteaza la corpul unei functii, fie sunt folosite pentru a se transmite anumite date auxiliare dintr-un modul in altul, dar acestea sunt nesemnificative (ca marime sau importanta) fata de cele deja enumerate.

**3.4. Formatul fisierelor utilizate**

Aplicatia trebuie sa functioneze pornind de la imaginile color și definitivand procesarea obținându-se PDF-uri bazate pe imaginile selectate.

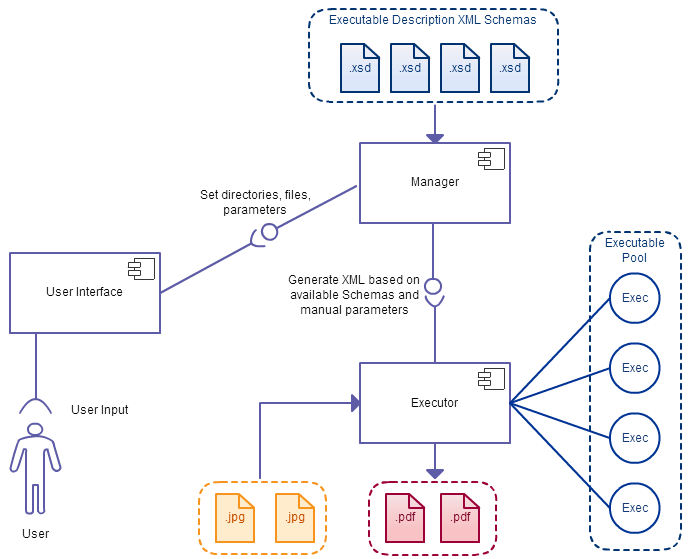
Pe parcursul procesarii vor fi folosite fisiere XML asociate cu schemele XML furnizate ,dar si toate executabilele ce pot fi descarcate de pe situl de cursuri <http://cs.curs.pub.ro/2012/mod/resource/view.php?id=1763>.

Executabilele sunt:

* preprocessing
* binarization
* layout
* paging
* OCR
* Hierarchy
* PDF-exporter

4. Modelul arhitectural si modelul componentelor

**4.1. Diagrama de arhitectura**



**4.2. Descrierea componentelor**

**1) User Interface**

Interfața grafică constă în partea de setup al unor configurări de baza, cum ar fi directoarele unde se află executabilele împreună cu schema xml a acestora și directorul de output.

Dupa ce setup-ul configurărilor este realizat de către utilizator și eventual validat de catre Document Processing Manager se vor selecta executabilele care se doresc a fi folosite (e.g. preprocessing, binarizarion, ocr) împreună cu parametrii și configurările specifice acestora.

**2) Document Manager Processing**

Manager-ul citește imaginile de input, aplică conform parametrilor configurați acțiuni asupra acestora prin rularea unor executabile cu ajutorul Executor-ului și are ca output niste fișiere .pdf.

Rolul Manager-ului este acela de a asigura comunicarea corectă între celelalte module pentru a obține rezultatele așteptate. Practic, el asigură un workflow corect al aplicației.

**3) Xml Creator**

Acesta asigură generarea de cod xml conform unor scheme xml puse la dispoziție in directorul setat in configurațiile menționate anterior.

Xml-ul obținut va fi necesar și pasat drept input unui executabil care asteaptă un fisier de tipul xml intr-o anumita forma de aceea este important ca scheme xml ale fiecarui fișier executabil sa fie disponibile și interpretate corect de către aceasta componentă.

**4) Executor**

Acesta componentă are rolul de a rula un executabil din Executor Pool. Este o componentă cu caracter generic, care ar trebui sa primeasca un fișier xml ca input, să îl valideze conform unei scheme și apoi sa ruleze un executabil.

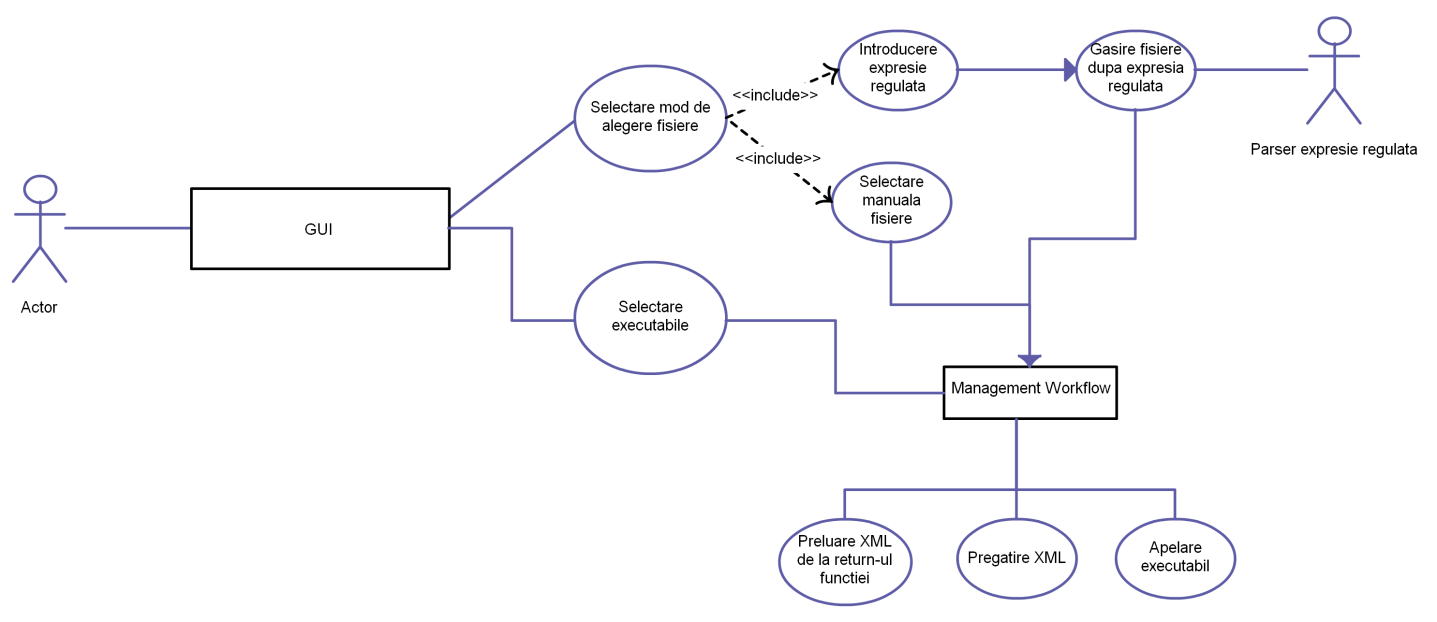
Executor-ul va returna chiar output-ul executabilului rulat.

**5) Executor Pool**

Executor Pool-ul este componenta cu care comunică Executor-ul atunci cand acesta trebuie sa ruleze. Practic, acesta asigură disponibilitatea fișierelor executabile Executor-ului.

**4.3. Restricţiile de implementare [TODO]**

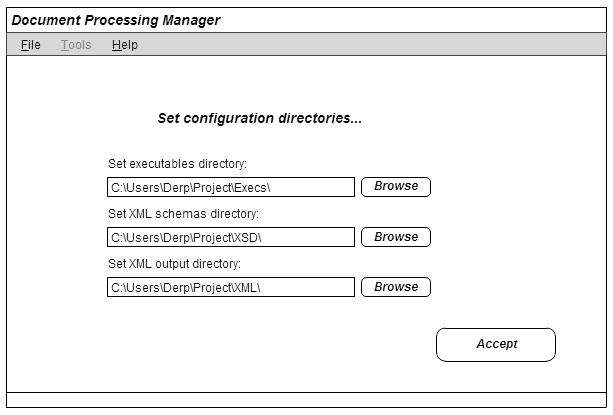
**4.4. Interacţiunea dintre component [TODO]**

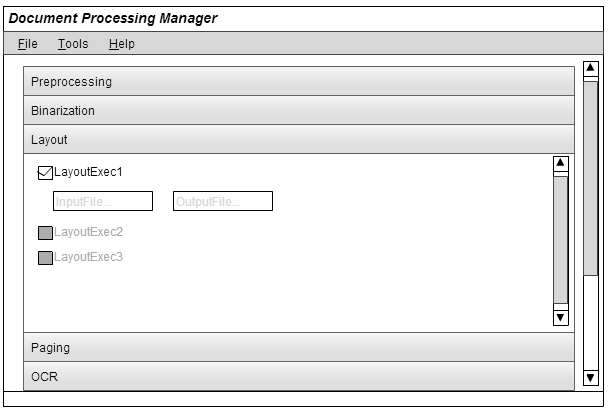


5. Modelul interfeţei cu utilizatorul

**5.1. Succesiunea interfeţelor [TODO]**

**5.2. Ferestrele aplicaţiei [TODO-incipient]**





6. Elemente de testare

**6.1. Componente critice [TODO]**

Fiecare modul va fi testat separat, in ordinea dezvoltarii aplicatiei .

**6.2. Alternative [TODO]**