

# Parseur / Validateur XML-Lite

Mathis Deloge, Antoine Petot, Ange Picard

05 octobre 2016

# Sommaire

- 1 Présentation du sujet
  - Le sujet
  - Prolongements possibles
  - Le XML-Lite
  - La structure du document
- 2 Modèle mathématique
- 3 Résultats
- 4 Présentation du programme
- 5 Conclusion

# Le sujet

## Le XML-Lite

Le XML est un langage de balisage générique très utilisé en informatique. Nous utiliserons une version simplifiée : le XML-Lite.

## Descriptif

Comme cité dans le sujet, un parseur / validateur XML-Lite est un programme capable de lire un fichier, d'indiquer s'il vérifie la norme XML-Lite et si oui, de l'analyser et de retenir sa structure ainsi que son contenu. L'utilisation d'un langage simplifié de XML nous permet de développer un programme réalisable dans le contexte donné tout en facilitant l'utilisation, les performances ainsi que les normes de conformité (XML 1.0).

# Prolongements possibles

## Les différents prolongements du sujet

- Permettre le débogage du fichier XML.
- Permettre au validateur de s'accorder à un schéma prédéfini.
- Modifier le validateur afin qu'il prenne en compte un schéma.
- Ajouter un interpréteur suivant le schéma d'un générateur de QCM.

## Les règles

- Une balise possède un nom.
- Une balise doit être ouverte puis fermée.
- Une balise peut contenir du texte.
- Une balise peut contenir d'autre balises.
- L'ordre des balises filles n'a pas d'importance et tout le texte contenu dans une balise est regroupé en un seul bloc.
- Une balise fille doit être fermée avant la fermeture de la balise parent.
- Une balise peut contenir une balise du même nom.
- Un document doit commencer par l'ouverture d'une balise se fermant à la fin du document.

# Exemples de XML-Lite

## XML-Lite correct

```
1 <FirstTag>
2     <ChildTag>
3         <AnotherChildTag>
4         </AnotherChildTag>
5     </ChildTag>
6     <tag>
7     </tag>
8 </FirstTag>
```

## Raisons

- Une balise s'ouvre en début de document et se ferme en fin.
- Respect de toutes les règles d'ouverture et fermeture de balise.
- Toutes les balises sont correctement imbriquées et nommées.

# Exemples de XML-Lite

## XML-Lite faux

```
1    <FirstTag>
2        <SecondTag>
3            <EndTag>
4                <AloneTag>
5                    </>
6        </FirstTag>
7    </SecondTag>
8    Un peu de texte
```

## Raisons

- Les balises ne sont pas correctement imbriquées.
- Une balise n'est pas correctement nommée.
- Une balise s'ouvre et ne se referme pas.

# La structure du document

## Descriptif

Le parseur / validateur doit être capable de lire n'importe quel fichier XML-Lite mais doit aussi être en mesure d'attendre une certaine structure de document grâce à l'ajout d'un fichier .dtd appelé schéma. Grâce aux fichiers schéma, le parseur / validateur connaît avec plus de finesse les balises filles autorisées ou non pour chaque balise. C'est une sorte de modèle qui permettra la validation du fichier XML-Lite.



# Sommaire

- 1 Présentation du sujet
- 2 **Modèle mathématique**
  - Choix du modèle mathématique
  - Représentation de l'automate fini
- 3 Résultats
- 4 Présentation du programme
- 5 Conclusion

# Choix du modèle mathématique

## Un automate fini

Pour nous permettre de parcourir rapidement un fichier XML-Lite, nous avons opté pour le développement d'un automate fini.

# Choix du modèle mathématique

## Un automate fini

Pour nous permettre de parcourir rapidement un fichier XML-Lite, nous avons opté pour le développement d'un automate fini.

## Simplicité et rigidité

Très peu d'états différents lors de la lecture d'un fichier XML-Lite. Les transitions entre états se font uniquement grâce à la différenciation des caractères '<', '>', '/' avec le reste.

# Choix du modèle mathématique

## Un automate fini

Pour nous permettre de parcourir rapidement un fichier XML-Lite, nous avons opté pour le développement d'un automate fini.

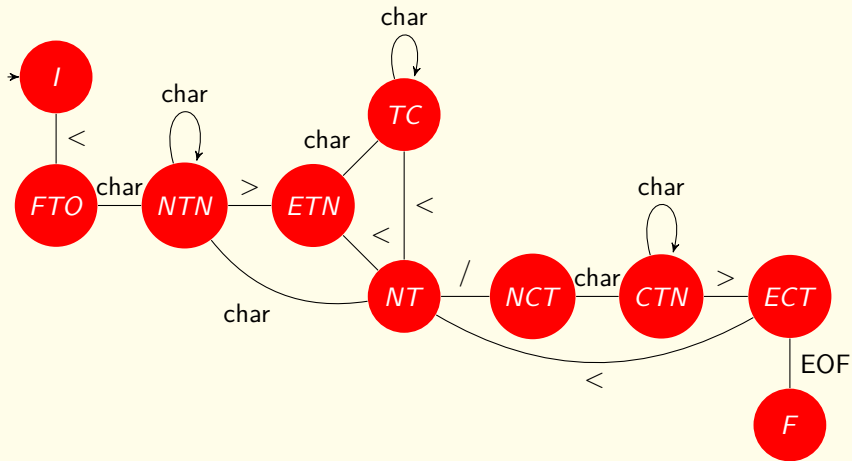
## Simplicité et rigidité

Très peu d'états différents lors de la lecture d'un fichier XML-Lite. Les transitions entre états se font uniquement grâce à la différenciation des caractères '<', '>', '/' avec le reste.

## Efficacité

Cette façon de parcourir un fichier XML-Lite caractère par caractère s'est avérée très rapide (exécution en 13ms pour un fichier XML-Lite de près de 700Mo).

# Représentation de l'automate fini



# Légende de l'automate

## Légende

I	Initial
FTO	First Tag Opening
NTN	New Tag Name
ETN	End Tag Name
TC	Text Content
NT	New Tag
NCT	New Closing Tag
CTN	Closing Tag Name
ECT	End Closing Tag
F	Final

# Sommaire

- 1 Présentation du sujet
- 2 Modèle mathématique
- 3 Résultats**
- 4 Présentation du programme
- 5 Conclusion

# Résultats



# Sommaire

- 1 Présentation du sujet
- 2 Modèle mathématique
- 3 Résultats
- 4 Présentation du programme
  - Le parseur
  - Le validateur
  - L'interpréteur
- 5 Conclusion

# Le parseur

## Parseur

```
1 package XMLLiteParser;  
2  
3 import XMLLiteParser.Gui.TreeView;  
4  
5 import javax.swing.tree.DefaultTreeModel;  
6 import javax.swing.tree.TreeNode;  
7 import java.io.IOException;  
8 import java.util.Date;
```

# Le validateur

## Validateur

```
1 public class main {
2     public static void main(String[] args)
3     {
4         TransitionSystem ts = new
5             TransitionSystem();
6         long startTime = System.
7             currentTimeMillis();
8         long elapsedTime;
9         try {
10             ts.parseFile("XMLDocs\\Ok.xml1
                ");
11         } catch (IOException e) {
12             e.printStackTrace();
13         }
14     }
15 }
```

# L'interpréteur

## Interpréteur

```
1
2      Node rootNode = Parser.getInstance
      ().getRootNode();
3
4      Schema schema = new Schema();
5      Constraint constraint = new
      Constraint("CD");
6      constraint.addChild(new Child("
      TITLE", true));
7      constraint.addChild(new Child("
      ARTIST", true));
8      constraint.addChild(new Child("
      COUNTRY", true));
9      constraint.addChild(new Child("
      COMPANY", true));
10     constraint.addChild(new Child("
      RELEASE", true));
```

# Sommaire

- 1 Présentation du sujet
- 2 Modèle mathématique
- 3 Résultats
- 4 Présentation du programme
- 5 Conclusion**

# Conclusion

## Difficultés rencontrées

- yolo

## Outils acquis

- yolo

# Sommaire

## 1 Présentation du sujet

- Le sujet
- Prolongements possibles
- Le XML-Lite
  - XML-Lite correct
  - XML-Lite faux
- La structure du document

## 2 Modèle mathématique

- Choix du modèle mathématique
- Représentation de l'automate fini

## 3 Résultats

## 4 Présentation du programme

- Le parseur
- Le validateur
- L'interpréteur

## 5 Conclusion