SER - Rapport Labo 1

Auteurs : * Da Silva Marques Fabio * Ganguillet Anne Sophie Catherine

Date: 19.03.2021

Introduction

Ce laboratoire a pour but de créer une DTD qui permet de modéliser la structure d'un document XML pour répondre a une certaine demande. Dans le cadre de ce laboratoire, la demande était de modéliser des tournois d'échecs pour une association fictive nommée "*Enjoy Chess*".

DTD

Pour voir la dtd veuillez vous référer au fichier en annexe : `echecs.dtd`

Explication des choix "simples"

Pour le choix de la mise en forme du contenu, nous avons décidés de créer des balises pour tout ce qui concerne les informations de type logique informatique en tant qu'attribut et tout le reste sous forme de balises. Le problème de cette approche est qu'on ne pourra pas vérifier le type de valeurs sous forme de text, mais cela nous permet de mieux structurer les données et d'améliorer la lecture et la comprehension des documents générés.

En ce qui concerne la validation des différentes balises et attributs nous avons inclus les dépendances simples mentionnées dans la donnée. Exemple : Un arbitre a un nom et un prénom => `<!ELEMENT Arbitre (Nom, Prenom)>`

Pour les données qui doivent avoir des valeurs précises, par exemple les pièces, nous les avons mis dans des balises, ce qui permet de faire une certaine vérification sur l'ensemble accepté de données.

Explication des choix "complexes"

Gestion Équipes

Pour la gestion des équipes nous avons les contraintes suivantes : * une partie est jouée entre 2 équipes * chaque équipe a 2 joueurs * chaque membre de l'équipe joue sur un échiquier différent * chaque équipe joue sur les deux camps

Pour résoudre ces contraintes notre approche a été la suivante.

En ce qui concerne la formation des équipes, elles sont définies à l'intérieur de chaque tournois étant donné qu'elles peuvent jouer plusieurs parties. Nous avons considéré que les équipes n'avaient pas leur place au niveau de l'organisation elle-même car nous pensons qu'elles ne seraient pas gardées d'un tournoi à un autre.

Ensuite pour résoudre les 3 contraintes restantes, nous avons choisi de placer une balise JoueurBlanc et JoueurNoir sur chaque échiquier d'une partie. Explication : * Une partie est jouée entre 2 équipes : 2 équipes = 4 joueurs, ce qui nous fait 2 joueurs blancs et 2 joueurs noirs * chaque membre joue sur un échiquier différent : Pour une équipe on aurait le joueur 1 qui joue sur l'échiquier 1 en blanc, et le joueur 2 qui joue sur l'échiquier 2 en noir. * chaque équipe joue sur les deux camps : même exemple que ci-dessus

Avec cette approche on évite la redondance de devoir reporter les équipes dans chaque partie étant donné qu'on gère seulement les références sur les identifiants des joueurs.

Gestion du score

Pour ce faire, nous avons décidé de mettre des balises qui référencent les 2 équipes de la partie en mettant le score en tant que contenu de la balise.

Gestion des coups

Premier point de logique on peut voir qu'il y a deux catégories distinctes de coups : * les déplacements : ce type de coups a une logique interne complexe * Les Roque : dont la seule information qu'il contient c'est s'il s'agit un `Grand Roque` ou `Petit Roque`

Ceci nous permet de faire un premier tri en mettant qu'un coup contient soit un `<Deplacement>` soit un `<Roque>`

Ensuite si on s'intéresse aux déplacements on a les informations suivantes : 1. La piece qui a été jouée 2. La case sur laquelle la pièce c'est rendue 3. La case depuis laquelle la pièce est partie 4. La pièce éliminée s'il y en a une 5. Une situation spéciale s'il y en a une

Pour les informations 2 a 5 il s'agit d'une modélisation simple comme explique dans le point `Explication des choix "simples"` donc on ne va pas s'attarder là-dessus.

En revanche en ce qui concerne la pièce qui a été jouée on a encore un cas complexe à résoudre. En effet selon la pièce qui est jouée on peut s'attendre a un comportement différent c'est pourquoi on a décidé de créer un objet qui correspond à chaque piece ce qui fait qu'un déplacement n'aura pas une balise pièce mais une balise de type `(Tour|Cavalier|Fou|Dame|Roi|Pion)`. De cette façon on permet de gérer pour chaque pièce sont comportement spécifique.

Par exemple pour le Pion nous avons la possibilité d'avoir une promotion qui peut être facilement modélisée à l'aide de cette approche.

Gestion du vainqueur du tournois

Pour cette partie on a décidé de mettre une balise `Vainqueur` qui référence simplement l'équipe qui a remporté le tournoi.

XML

Voir fichier en annexe: `echecs.xml`

Validation

validation de la dtd

Validation faite avec https://www.xmlvalidation.com/index.php?id=1&L=0

Réponse aux questions

```
Imaginons que vous souhaitez enregistrer le classement ELO que chaque joueur d'une partie avait au moment où elle a
été jouée, qu'est-ce qu'il faudrait modifier dans votre DTD ?
```

> Il faudrait modifier les balises JoueurBlanc et JoueurNoir de la sorte :
>
> ```
> <JoueurBlanc idJoueur="J1">
>   <!-- Ici la nouvelle balise -->
>   <ELO> 1200 </ELO>
> </JoueurBlanc>
> ```

```
Il est possible dans votre DTD de représenter le fait qu'il ne peut y avoir que 20 parties par tournoi.
- Comment faire ? Discutez également de votre solution à ce problème.
```

> Une méthode consisterait à modifier l'élément `Parties` en y mettant 20 fois un contenu de type Partie :
>
> ```
> <!ELEMENT Parties (Partie,Partie?,Partie?,Partie?,...)>
> ```

```
Est-ce possible dans votre DTD d'interdire le fait qu'une équipe joue contre elle-même dans une partie ?
Justifiez votre réponse.
```

> Non parce qu'il n'est pas possible de vérifier l'exactitude des données avec une DTD

```
Est-ce possible dans votre DTD de vérifier que pour une partie, l'arbitre fait bien partie du tournoi dans lequel la
partie en question est jouée ? Justifiez votre réponse
```

> La DTD ne peut pas faire cette vérification, on peut vérifier que l'arbitre existe mais on ne peut pas vérifier qu'il participe au tournois.

Conclusion

Au terme de ce laboratoire nous avons réussi à modéliser toutes les contraintes qui nous semblaient possibles dans notre DTD. On a ainsi pu constater les limites du "langage".

Après avoir terminé ce laboratoire, on se rend compte que la DTD ne sert pas du tout a vérifier l'exactitude des données mais plutôt à garantir qu'une certaine structure est respectée dans le document XML. Ça permet néanmoins de faciliter une partie du travail de vérification pour les outils qui devront aller lire les données dans ces fichiers, étant donné qu'il ne reste plus qu'à vérifier l'exactitude des données.