Académie de Montpellier
Université Montpellier II
Sciences et Techniques du Languedoc

MÉMOIRE DE

STAGE RECHERCHE MASTER M2

effectuée au Laboratoire d'Informatique de Robotique
et de Micro-électronique de Montpellier

Spécialité : **AIGLE**

Personnalisation de page web : application à l'amélioration de l'accessibilité du web

par **franck PETITDEMANGE**

Sous la direction de **Marianne HUCHARD, Michel MEYNARD, Yoann BONAVERO**

Table des matières

1 Méta-modèle de page web

1.1 HTML 4

HTML 4 [?] est un langage permettant la publication de contenu sur le web. C'est le langage standard actuel des pages web. Il permet de structurer le contenu et de lui associer une mise en forme. Le contenu peut être du texte, des images, ou plus généralement du multimédia. Ce contenu est organisé de manière hiérarchique en le découpant en section et sous-section.

Contenu Le contenu principal décrit dans les pages HTML 4 est un contenu textuel. Il peut également contenir du multimédia comme des images, des vidéos et applets (des programmes qui sont automatiquement chargés puis lancés sur la machine de l'utilisateur). L'inclusion de contenu multimédia se fait par l'élément générique : <OBJECT>. Il possède une collection d'attributs prédéfinis qui décrivent l'objet inclus dans la page. Le principal étant *type* décrivant le type de contenu des données (*e.g.* figure 1). La valeur de ces attributs n'est pas prédéfinie. Elle est interprétée librement par la machine qui charge la page web.

```
<object data="data/test.mpg" type="video/mpeg">
        Ceci est une vidéo
</object>
```

FIGURE 1 – Exemple contenu multimédia

Structuration générique HTML 4 propose un mécanisme générique pour la composition de contenu formant la structure des pages web. Ce mécanisme gravite autour des éléments de type <DIV> et de leurs attributs respectifs : *id* et *class*.

DIV Signifiant division, la balise DIV est utilisée comme conteneur générique, il peut contenir n'importe quel élément. Il est exploité pour :
- regrouper les éléments pour leur appliquer un style (une mise en forme particulière).
- signaler une section ou une sous-section.

id et class Chaque élément peut se voir attribuer un identifiant ou une classe d'appartenance. *id* assigne un nom à un élément. Ce nom est unique dans le document. *class* au contraire, assigne un ou plusieurs noms de classe à un élément. Un nom de classe peut être partagé par plusieurs instances d'éléments. Les identifiants et les classes sont des suites de caractères quelconques décidées arbitrairement par l'auteur du document.

Les éléments DIV utilisés conjointement avec les attributs *id* et *class* sont au cœur du mécanisme générique de structuration d'un document. DIV permet de diviser le contenu d'un document en sections et sous-sections (*e.g.* figure 3) pour décrire sa structure. Les balises <DIV> ayant une sémantique neutre, c'est l'auteur du contenu qui attribue (de manière arbitraire) un nom de *class* ou un *id*. L'*id* ou la *class* est associé à une mise en forme définie a priori. La mise en forme est définie au travers d'un langage : CSS[?] que l'on appelle feuille de style. CSS permet d'appliquer un ensemble de règles de style ou un agencement des éléments dans l'espace de la page. Par exemple,

l'auteur peut déclarer une classe "aside" et définir que les éléments appartenant à la classe "aside" doivent être placés sur le côté droit de la page avec un fond blanc. Ce mécanisme est illustré par la figure 2. L'auteur associe à chaque <DIV> une *class* ou un *id* auquel s'applique une mise en page et une mise en forme définies par l'auteur dans une feuille de style CSS.

Méta-modèle La figure 4 modélise les éléments principaux de construction d'une page web suivant la spécification de HTML 4.

- Chaque élément hérite de la méta-classe **Element**. Cela permet d'associer un identifiant unique à chaque élément ainsi qu'un ensemble de classe.
- Les éléments de la méta-classe **Bloc** sont l'ensemble des éléments dans une page qui forment un bloc structurel (*e.g.* un paragraphe).
- Les éléments de la méta-classe **InLine** définissent des éléments qui ne forment pas de blocs structurels. Par exemple la balise *strong* ne définit pas un bloc structurel de la page mais indique que l'élément qu'elle encapsule est un mot important dans un texte. De manière générale les éléments *InLine* servent à attribuer une sémantique aux éléments textuels.
- La relation de composition entre les éléments *Bloc*, spécifie que les éléments *Bloc* peuvent contenir des éléments *InLine* mais pas l'inverse. La relation de composition réflexive de l'élément *Bloc* spécifie qu'un élément en bloc peut contenir d'autre élément *Bloc*, il en est de même pour les éléments *InLine*.
- Les éléments de la méta-classe **Header** spécifient des éléments de titre. Ils introduisent le titre d'une section ou sous-section.
- Les éléments de la méta-classe **Div** spécifient une division structurelle : une section, sous section. En pratique elle est également utilisée pour signaler des regroupements d'élément afin de leurs appliquer une mise en forme.
- Les éléments de la méta-classe **Paragraph** forment une composition logique d'éléments textuels
- Les éléments de la méta-classe **Object** spécifient l'inclusion d'un contenu multimédia ou un programme.

1.2 HTML 5

HTML 5 [**?**] étend HTML 4 en apportant de nouveaux éléments lexicaux. Ces nouveaux éléments apportent une sémantique standard et de plus haut niveau. Elle permet notamment d'expliciter la structure d'une page.

Contenu HTML 5 fournit de nouveaux éléments comme <VIDEO>, <AUDIO> avec un ensemble d'attributs propres à chaque balise (a contrario de l'élément <OBJECT> de HTML 4). Les attributs spécifiques permettent de renseigner l'état d'un élément. Par exemple, la balise <AUDIO> possède un attribut spécifique *muted* indiquant si le son de l'élément audio est coupé ou non.

Structuration Les nouveaux éléments de HTML 5 spécifient donc une sémantique standard :

- SECTION : représente une section générique dans un document, c'est-à-dire un regroupement de contenu par thématique.
- ARTICLE : représente un contenu autonome dans une page, facilite l'inclusion de plusieurs sous-documents.
- NAV : représente une section de liens vers d'autres pages ou des fragments de cette page

- ASIDE : représente une section de la page dont le contenu est indirectement lié à ce qui l'entoure et qui pourrait être séparé de cet environnement
- HEADER : représente un groupe d'introduction ou une aide à la navigation. Il peut contenir des éléments de titre, mais aussi d'autres éléments tels qu'un logo, un formulaire de recherche, etc.
- FOOTER : représente le pied de page, ou de la section, ou de la racine de sectionnement la plus proche

La figure 5 montre un découpage explicite de la structure avec HTML 5 en opposition au découpage implicite de HTML 4 montré dans la figure 2.

Méta-modèle La figure 6 modélise les concepts principaux de construction d'une page web avec HTML 5.

- La méta-classe **Sectioning** définit le contenu comme des éléments qui créent une nouvelle section dans le plan d'un document. Ils définissent également la portée des éléments d' en-tête (Header) et de pied de page (Footer). Elle encapsule les concepts de division de HTML 4 (<DIV>).
- La méta-classe **Header** définit le contenu comme des éléments d'introduction. Par exemple pour un page web, un logo ou pour une section, un titre. Les sous classe de Header sont des éléments de titre introduisant des sections, ce sont les mêmes concepts de titre que pour HTML 4.
- La méta-classe **Footer** définit le contenu comme étant des éléments de pied de page ou de section.
- La méta-classe **Phrasing** définit le contenu textuelle, elle encapsule le concept de balise *Inline* de HTML 4.
- La méta-classe **Embedded** définit un contenu importé dans une page. C'est le cas par exemple des éléments de type <video> qui sont sous classe de la méta-classe Embedded (elle est également sous-classe de la méta-classe Interactive). Elle encapsule les concepts de *Object* de HTML 4.
- La méta-classe **Interactive** définit le contenu interactif dans une page. Par exemple, la balise <a> qui définit une navigation vers une autre ressource.

1.3 ARIA

ARIA (Acessible Rich Internet Application) [**?**] est la spécification d'une Ontologie décrivant une interface graphique. Elle fournit des informations sur la structuration d'un document et plus généralement elle décrit les éléments qui composent une interface au moyen d'un ensemble de rôles, d'états et de propriétés.

Rôle Les rôles permettent d'identifier la fonction de chaque élément d'une interface. Ils sont regroupés en trois catégories :

- Widget Roles : définit un ensemble de widget (alertdialog, button, slider, scrollbar, menu, etc.)
- Document Structure Roles : décrit les structures qui organisent un document (article, definition, entête, ect.)
- Landmark Roles : décrit les régions principales d'une interface graphique (main, navigation, search, etc.)

États et propriétés ARIA prend en compte l'aspect dynamique et interactif des éléments d'une interface. Elle permet d'associer des états et des propriétés aux éléments d'une interface. Un état est une configuration unique d'un objet. Par exemple, on peut définir l'état d'un bouton par l'état *aria-checked* qui peut prendre trois propriétés suivant l'interaction avec l'utilisateur : *true* - *false* - *mixed* . Dans le cas d'une checkbox, *true* indique si la checkbox est cochée, *false* si elle ne l'est pas et *mixed* dans le cas d'un ensemble de checkbox indique que certaines sont cochées.

Aria prévoit même un système d'annotation pour les objets ayant des comportements asynchrones. Par exemple, on peut indiquer par une annotation qu'un élément se met à jour de manière autonome.

Méta-modèle La figure **??** modélise les principales méta-classe de la norme ARIA. On distingue deux groupes. Un premier groupe pour la description des éléments d'interactions qui hérite de la méta-classe **Widget** :

- La méta-classe **Composite** indique qu'un élément graphique possède des éléments navigables. Elle permet d'organiser la navigation à l' intérieure d'un *Wigdet*.
- La méta-classe **Input** définit des éléments qui permettent des saisies de la part d'un utilisateur.
- La méta-classe **Command** définit des éléments qui réalisent des actions. Par exemple, l'envoie de données vers un serveur.

Un deuxième groupe pour les éléments de structuration qui héritent de la méta-classe **Structure**

- La méta-classe **Section** définit les éléments qui définissent une unité de confinement structurelle dans une page.
- La méta-classe **Sectionhead** définit les éléments qui introduisent des titres.
- La méta-classe **Landmark** définit les principaux points d ?intérêts dans une page. Par exemple, la banniere d'une page, les formulaires, le contenu principal.

1.4 CSS

CSS est un langage de feuille de style qui permet aux auteurs des pages web de lier du style aux éléments HTML. Le style définit comment afficher un élément (ex. les polices de caractères, l'espacement, couleurs, *etc.*). CSS permet ainsi de séparer la présentation du style du contenu (*cf.* figure 7). L'avantage est une simplification de l'édition et de la maintenance d'une page web.

Modèle de boîte CSS génère pour chaque élément de l'arbre du document (DOM) une boîte rectangulaire. Les boîtes rectangulaire sont conformes à un modèle de boîte et sont agencées suivant un modèle de mise en forme décrit en section 1.4

Chaque boîte possède ainsi une aire de contenu (*e.g* une texte, une image, *etc.*) entourée en option par une aire d'espacement, une aire de bordure et une aire de marge (*e.g* figure 8).

Modèle de mise en forme Chaque boîte se voit attribuer un type qui affecte en partie son positionnement. Les deux principaux types sont les *boîtes en bloc* et *les boîte en-ligne*. Les éléments de type bloc sont des éléments dont le rendu visuel forme un bloc (*e.g* figure 9 avec l'élément de paragraphe <p>). Les éléments de type en-ligne sont des éléments qui n'ont pas la forme de blocs de contenu (*e.g* figure 9 avec l'élément). Les boîtes en-ligne sont placées horizontalement, l'une après l'autre, en commençant en haut du bloc conteneur. Les blocs conteneurs sont des boîtes qui encapsulent d'autres boîtes. Les boîtes en-bloc sont placées l'un après l'autre, verticalement, en commençant en haut du bloc conteneur. Le schéma de positionnement décrit est appelé *flux normal*.

Une fois le *flux normal* calculé, il est possible de le modifier.

Un premier mécanisme possible est le positionnement relatif. La position de la boîte est exprimée en propriété de décalage par rapport à son bloc conteneur :

- 'top' : définit le décalage du bord haut de la marge d'une boîte sous le bord haut de la boîte du bloc conteneur.
- 'right' : définit le décalage du bord droit de la marge d'une boîte à gauche du bord droit de la boîte du bloc conteneur.
- 'bottom' : définit le décalage du bord bas de la marge d'une boîte au-dessus du bord bas de la boîte du bloc conteneur.
- 'left' : définit le décalage du bord gauche de la marge d'une boîte à droite du bord gauche de la boîte du bloc conteneur.

Un deuxieme mécanisme est le positionnement flottant. Une boîte flottante est déplacée vers la gauche ou la droite sur la ligne courante du *flux normal*. Le contenu du document s'écoule alors le long des flancs de cette dernière.

Un troisième mécanisme est le positionnement absolu. La boîte est retirée du *flux normal* et est positionnée par rapport à son bloc conteneur. La différence avec le positionnement relatif est que le positionnement de la boîte n'a aucun effet sur les boîtes du même niveau de parenté. Ces boîtes peuvent, ou non, cacher les autres boîte.

Avant-plan et d'arrière-plan Les propriétés CSS permettent aux auteurs la spécification d'une couleur d'avant-plan et d'arrière-plan pour un élément. La couleur d'arrière-plan peut être une couleur ou une image . L'arrière-plan correspond aux aires de contenu et, d'espacement et de bordure. Le couleur d'avant-plan correspond à la couleur du contenu de texte d'un élément.

Les polices CSS permet de pouvoir spécifier l'utilisation de plusieurs représentation pour les caractères textuelles : *la police*. Une liste exaustive de propriétés permettent de spécifier la police d'un élément contenu dans une boîte. On peut spécifier par exemple une famille de police (serif, sans-serif), le style de la police (italic, oblique), la taille, *ect.*

Les textes CSS définies la représentation visuelle des caractères, des caractères blancs, des mots et des paragraphes. On peut spécifié un alinéa pour la première ligne du texte dans un bloc ('text-indent'), l'alignement d'un contenu en-ligne dans un élément de type bloc ('text-align'), le comportement de l'espacement entre les caractères du texte ('letter-spacing'), *ect.*

2 Réalisation

2.1 Introduction

La conception du méta-modèle est double. S'affranchir de la diversité de représentation des onnées, support à,l'expression des souhaits, La conception du modèle de contenu associe une sémantique aux structures syntaxiques des langage de publication de contenu.

2.2 Méta-modèle

2.2.1 Modèle de contenu

Relation de composition ? En raison de la grande souplesse du langage, on ne fait pas apparaître le relation de composition dans le modèle de contenu, mais plutôt dans le modèle de mise en forme. La spécification de composition des balises en ligne et en bloc est annulé par les possibilités de CSS permettent de modifier la nature en faisant passé une balise de en bloc a en ligne.

2.2.2 Modèle de mise en forme

L'étude des spécification de la norme CSS nous a amené à la réalisation d'un méta-modèle (*cf.* figure 11) restreint aux aspects des principaux éléments de présentation conforme aux souhaits de transformation. Les spécifications de mise en forme n'apparaîssent pas dans le méta-modèle, il n'est pas prévu d'integrer pour la durée du stage l'expression des besoins sur l'agencement des éléments. Les éléments les plus important sont les couleurs d'arrière plan et du texte qui seront le support à l'expression des souhaits constrastes (de couleurs) ainsi qu'au module de calcule.

Méta-classe Box La méta-classe *Box* décrit le concept de bloc conteneur au travers de la relation de composition. La propriété Display sert à définir le type de boîte (*inline* et *block*). Ce genre de propriété pourrait nous permettre l'expression du besoin d'afficher un menu horizontalement ou vericalement. La propriété Margin spécifie l'espacement du bord extérieur de la boîte. Elle nous permettra de mieu marquer la séparation entre les différents types de contenu.

Méta-classe Style La méta-classe *Style* des boîtes de CSS décrit les boîtes rectangulaires qui sont générées pour les éléments de l'arbre du document (*cf.* figure 8). La méta-classe *Style* décrit :
- padding : l'air d'espacement,
- border-width épaisseur de bordure,
- *border-style* : style de la bordure,
- border-color : la couleur de bordure,
- border-[color, image] : arière-plan .

Méta-classe Texte La méta-classe *Texte* décrit la répresentation visuelle des caractères, mots et paragraphes contenuent dans une boîte. Par exemple, la propriété *letter-spacing* spécifie l'espace entre les lettres :
- text-indent : décrit un alinea
- text-align : décrit un alignement. Exemple de valeur possible : alignement de texte à gauche, droite, centré, *ect.*
- decoration : décrit un trait en-dessous, trait au-dessus, rayure et clignotement
- text-shadow : décrit des effets d'ombrage appliquer au texte
- letter-spacing : décrit l'espacement entre les mots
- word-spacing : décrit l'espacement entre les mots
- text-transform : décrit les effets de capitalisation dans le texte. Par exemple la valeur *uppercase* définit que les lettres de chaque mots soient en majuscule, *lowercase* décrit l'inverse.
- color : décrit la couleur du texte

Méta-classe Police La méta-classe *Police* décrit la représentation visuelle des caractères :

- font-family : décrit le noms de familles génériques de la police du texte (*e.g new century schoolbook*)
- font-style : style de la police (*e.g italic*)
- font-weight : décrit la graisse de la police
- font-size : décrit la taille de la police

Glossaire

Ontologie En philosophie, l'ontologie est l'étude de l'être en tant qu'être, c'est-à-dire l'étude des propriétés générales de ce qui existe. Par analogie, le terme est repris en informatique et en science de l'information, où une ontologie est l'ensemble structuré des termes et concepts représentant le sens d'un champ d'informations [?]. 5

```
<body>
        <div id="header" ></div>
        <div id="navigation_bar"></div>
        <div class="aside"></div>
        <div class="section">
                <div class="article"></div>
                <div class="article"></div>
        </div>
        <div class="aside"></div>
        <div id="footer"></div>
</body>
```

FIGURE 2 – Architecture page web HTML 4

```
<body>
<div class="section" id="elephants-foret" >
        <h1>Les éléphants des forêts</h1>
        <p>Dans cette partie, nous abordons le sujet
moins connu des éléphants des forêts.</p>
        <div class="sous-section" id="habitat-foret" >
        <h2>L'habitat</h2>
        <p>Les éléphants des forêts ne vivent pas
dans les arbres mais au milieu d'eux.</p>
        </div>
</div>
</body>
```

FIGURE 3 – Exemple découpage en sections et sous-sections

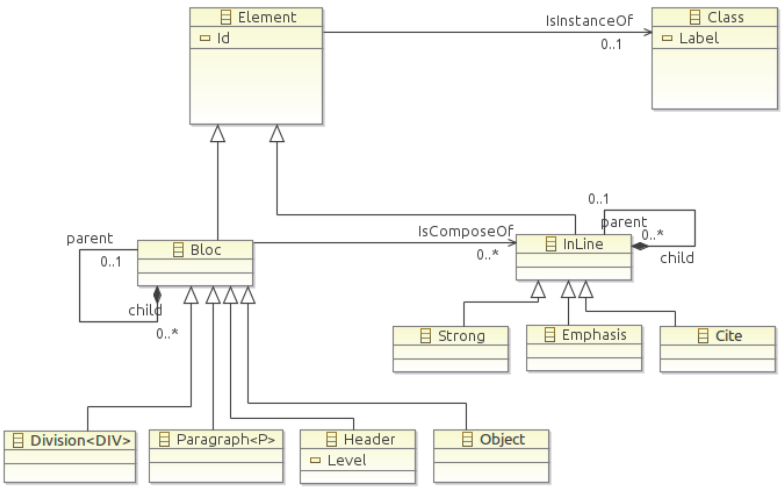

FIGURE 4 – Méta-modèle HTML 4

```
<body>
        <header></header>
        <nav></nav>
        <section>
                <article></article>
                <article></article>
        </section>
        <aside></aside>
        <footer></footer>
</body>
```

FIGURE 5 – Exemple d'attribution de rôle

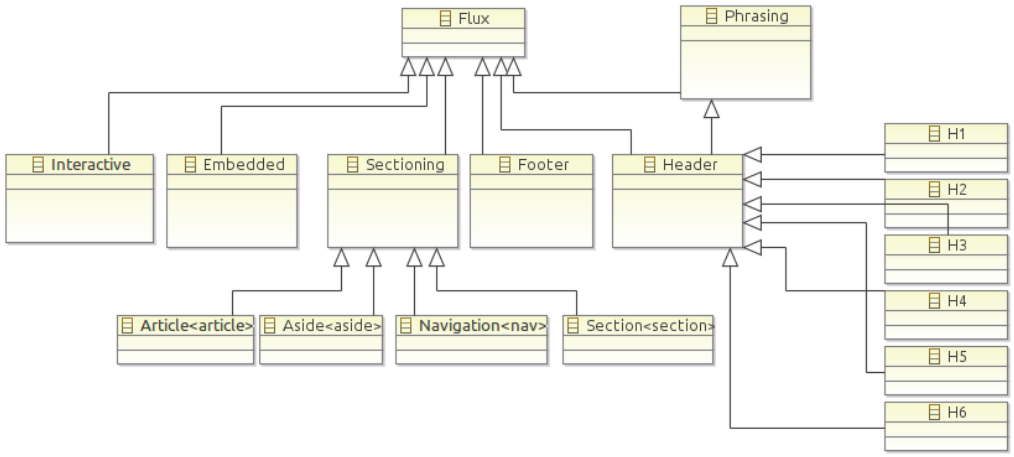

FIGURE 6 – Méta-modèle HTML 5

```
<style>
p.serif{font-family:"Times New Roman",Times,serif;}
p.sansserif{font-family:Arial,Helvetica,sans-serif;}
</style>

<body>
<p class="serif">Ceci est un paragraphe
avec un style de font Times New Roman font.</p>
<p class="sansserif">Ceci est un paragraphe
avec un style de font the Arial font.</p>
</body>
</html>
```

Ceci est un paragraphe avec un style de font Times New Roman font.

Ceci est un paragraphe avec un style de font the Arial font.

FIGURE 7 – Exemple CSS

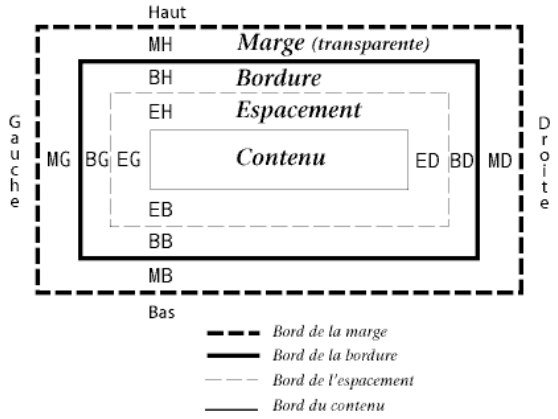

FIGURE 8 – Modèle de boîte

```
<p>Avant de faire le truc X il est
<strong>nécessaire</strong> de faire le truc Y avant.
</p>
```

FIGURE 9 – Exemple élément en-line

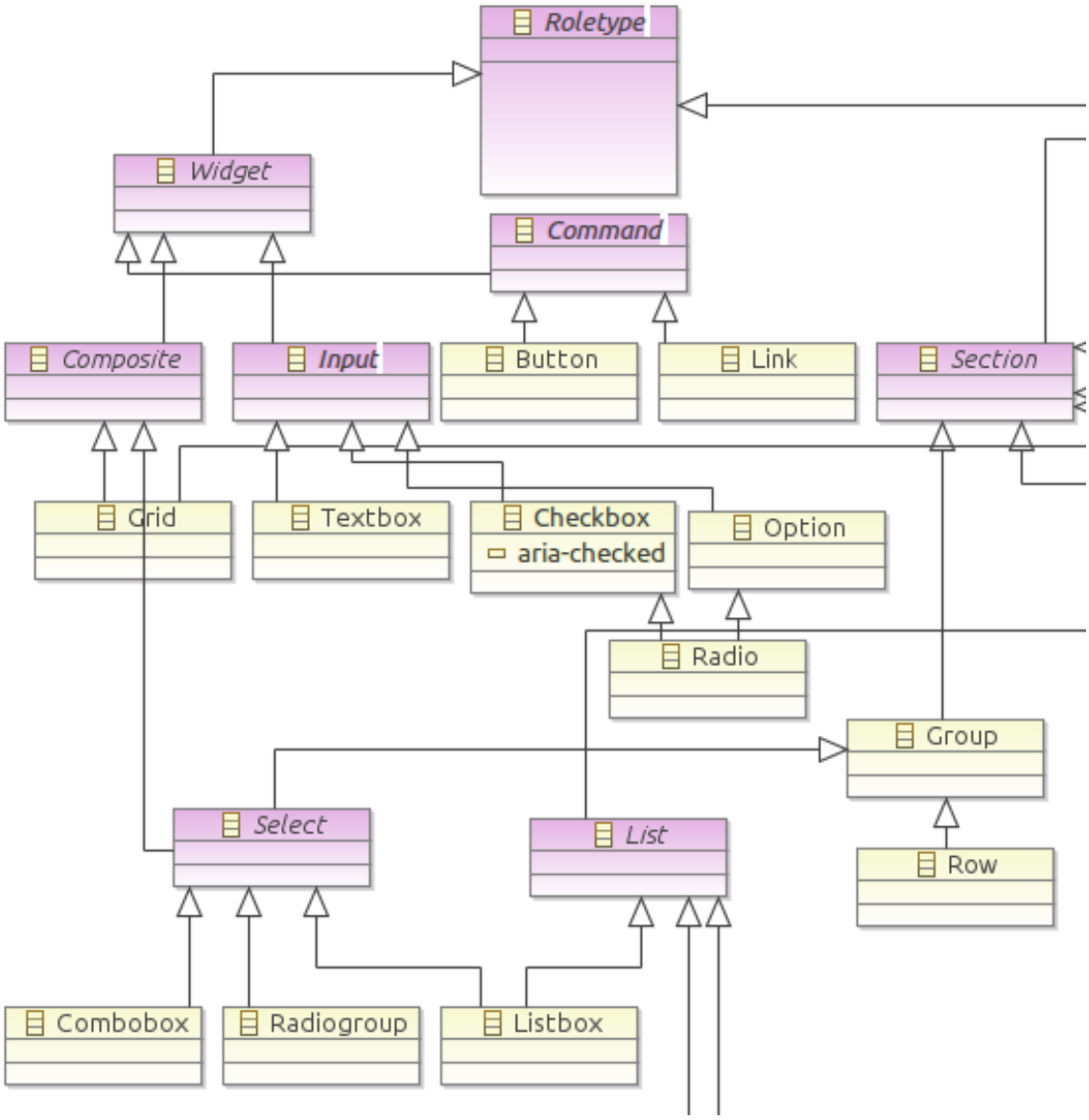

FIGURE 10 – Méta-modèle

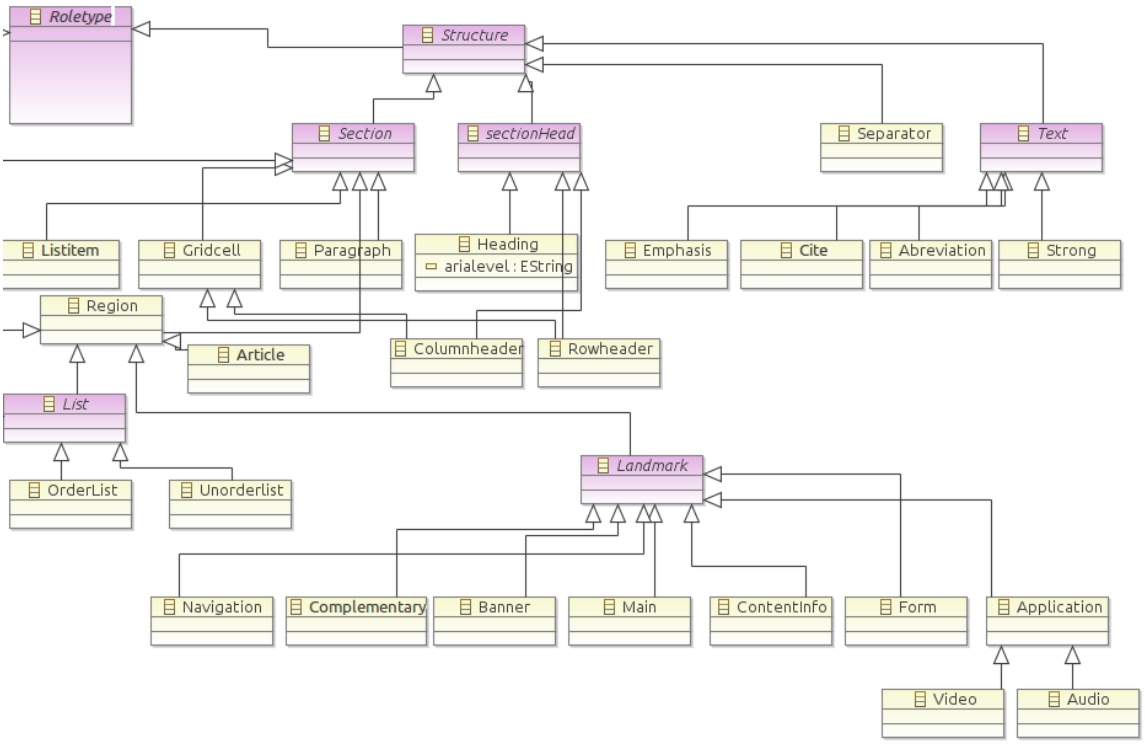

FIGURE 11 – Méta-modèle

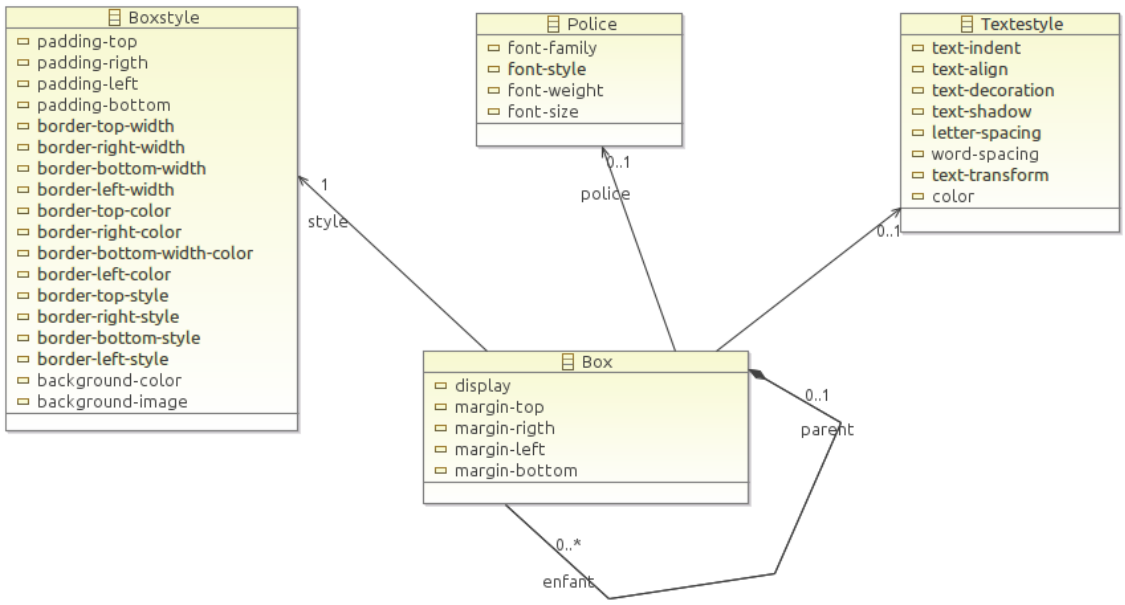

FIGURE 12 – Méta-modèle