À partir d'une image numérique, des modèles HTR produisent une transcription de la page en format XML ALTO (Analyzed Layout and Text Object). Comme expliqué et justifié dans le chapitre **??**, ce format est transformé en le format *pivot* du pipeline *Gallic(orpor)a*, la TEI (Text Encoding Initiative). Le schème TEI est flexible et il permet d'encoder la transcription d'un document texte en plusieurs façons. La manière préférée par l'équipe du projet *Gallic(orpor)a* s'appuie sur l'élément TEI `<sourceDoc>`. Selon les *guidelines* de la Text Encoding Initiative, l'élément `<sourceDoc>` contient *a transcription or other representation of a single source document potentially forming part of a dossier génétique or collection of sources*[1]. L'autre option est l'élément `<facsimile>`. Il contient *a representation of some written source in the form of a set of images rather than as transcribed or encoded text*[2].

Entre les deux options, l'élément `<souceDoc>` convient mieux à la transcription encodée dans un fichier ALTO puisqu'il est destiné à traiter la représentation d'une source. Un fichier ALTO contient la représentation d'une image de texte. L'élément `<facsimile>` pourrait bien servir à l'encodage de l'image elle-même, mais sa représentation dans le fichier ALTO est mieux encodée avec l'élément `<sourceDoc>`. Après tout, il faut se souvenir que le pipeline *Gallic(orpor)a* vise à conserver dans le document TEI toute donnée de la transcription du fichier ALTO. L'un des objectifs du projet est que la ressource numérique produite par le pipeline permet de recréer des fichiers ALTO à partir du document TEI, afin que des utilisateurs puissent utiliser les fichiers ALTO reconstruits comme des vérités de terrain et entraîner des nouveaux modèles HTR. Il faut donc un élément TEI destiné à la transcription d'une image, au lieu de l'image elle-même.

Par rapport au *mapping* des données du `<teiHeader>`, le *mapping* des données du `<sourceDoc>` est plus direct et fixé. Tandis que le contenu du `<teiHeader>` compte sur la disponibilité variable des données depuis les divers sources de données, le contenu du `<sourceDoc>` est très prévisible et compte sur un schème ALTO qui est très systématique. La manière de transcrire les pages numérisées ne change pas et s'effectue par les mêmes modèles HTR bien que les documents transcrits soient différents de l'un à l'autre.

Cependant, il y a une variation possible dans la granularité de la transcription. D'une part, le fichier ALTO peut porter des prédictions sur les caractères d'un mot, y compris leur contenu et leur emplacement sur la page. De l'autre part, le fichier ALTO peut porter des prédictions sur une ligne de texte, où son contenu est une chaîne des mots et des espaces entre mots. Dans ce dernier cas plus simple, le fichier ALTO présente moins de détail par rapport à l'autre forme. L'application `alto2tei` s'adapte aux deux puisqu'elles sont toutes les deux valables et produites par l'engin *Kraken*. Depuis la ligne de commande, l'engin *Kraken* met en pratique des modèles HTR et produit les fichiers ALTO qui contiennent des prédictions sur les caractères ou les « glyphes » des mots de la ligne de texte. Depuis l'interface *eScriptorium*, la même version de *Kraken* et

1. *TEI element sourceDoc*. URL : `https://tei-c.org/release/doc/tei-p5-doc/en/html/ref-sourceDoc.html` (visité le 23/08/2022).
2. *TEI element facsimile*. URL : `https://tei-c.org/release/doc/tei-p5-doc/en/html/ref-facsimile.html` (visité le 03/09/2022).

les mêmes modèles HTR produisent des fichiers ALTO qui ne contiennent que des prédictions sur la ligne de texte, n'allant pas en détail sur les caractères et les mots ou les « segments ». Les formes différentes sont expliquées dans la section **??** du chapitre **??**.

1 Le modèle du `<sourceDoc>`

L'application `alto2tei` récupère toute donnée significative des fichiers ALTO et les met sur un arbre TEI qu'elle construit, spécifiquement sur la branche de l'élément `<sourceDoc>`. L'arborescence du `<sourceDoc>` prendre deux formes, selon le détail du fichier ALTO que des modèles HTR ont produit. Si les modèles avaient enregistré des prédictions sur les glyphes et les segments dans le fichier ALTO, le `<sourceDoc>` aurait quatre niveaux d'élément `<zone>` pour porter sur les masques d'un bloc de texte, d'une ligne dans le bloc, d'un mot dans la ligne, et d'un caractère dans le mot. Le contenu textuel serait donc représenté deux fois ; il serait balisé dans l'élément qui porte sur la prédiction du caractère et aussi dans un élément qui représente tous caractères prédits dans une ligne de texte. L'exemple des quatre niveaux est montré dans la Figure 1 Si les modèles avaient enregistré uniquement des prédictions sur les blocs et les lignes de texte, le `<sourceDoc>` aurait deux niveaux d'élément `<zone>`, un pour représenter le bloc prédit et l'autre pour la ligne prédite. L'exemple de cette arborescence plus simple est montré dans la Figure 2.

```
<sourceDoc>
  <surface><!-- Région d'une page -->
<!-- ... -->
    <zone type="SegmOntoZone"><!-- Région d'un bloc de texte -->
<!-- ... -->
      <zone type="SegmOntoLine"><!-- Région d'une ligne de texte (ex. "
    Texte ici.") -->
<!-- ... -->
        <zone type="String"><!-- Région d'un segment dans la ligne (ex.
     "Texte") -->
<!-- ... -->
          <zone type="Glyph"><!-- Région d'un caractère dans le segment
     (ex. "T") -->
<!-- ... -->
            <c>T</c>
          </zone>
<!-- ... -->
        <zone type="Space"/><!-- Région d'une espace entre mots dans la
     ligne -->
<!-- ... -->
        <line>Texte ici.</line>
      </zone>
    </zone>
  </surface>
</sourceDoc>
```

FIGURE 1 – Le `<sourceDoc>` de quatre niveaux de masques imbriqués

```
<sourceDoc>
  <surface><!-- Région d'une page -->
<!-- ... -->
    <zone type="SegmOntoZone"><!-- Région d'un bloc de texte -->
<!-- ... -->
      <zone type="SegmOntoLine"><!-- Région d'une ligne de texte (ex. "Texte ici.") -->
        <line>Texte ici</line>
      </zone>
    </zone>
  </surface>
</sourceDoc>
```

FIGURE 2 – Le `<sourceDoc>` de deux niveaux de masques imbriqués

1.1 La page

Normalement, un document XML ALTO représente la transcription d'une seule page du document source. Dans le schème ALTO, des données portant sur une page sont imbriquées dans l'élément `<Page>` dont les attributs décrivent le longueur (`@WIDTH`) et l'hauteur (`@HEIGHT`), ainsi que le compte d'image (`@PHYSICAL_IMAGE_NR`) dans la suite des images traitées. Contrairement aux données susdites, il ne faut pas conserver l'identifiant (`@ID`) donné à la page. En reconstituant un fichier ALTO à partir du `<sourceDoc>`, peu important quel identifiant peut être donné à la nouvelle `<Page>` pourvu qu'il soit unique. L'application `alto2tei` génère un nouveau identifiant pour la `<Page>` et pour tout élément TEI qu'elle construit.

1.1.1 Des règles générales sur l'identifiant de l'élément TEI

L'identifiant de tout élément descendant du `<sourceDoc>` se construit de certains composants qui s'accumulent. Le parent, la page, porte tout simplement l'identifiant de la page, c'est-à-dire le numéro du folio. Tout élément descendant et imbriqué dans la page retient cette donnée dans son identifiant, en y ajoutant à la suite encore plus de données. Par exemple, l'identifiant de la page de l'onzième folio du fac-similé numérique serait « f11 ». L'identifiant du premier bloc du texte sur l'onzième folio porterait donc l'identifiant « f11-textblock_0-blockCount1 ». L'identifiant du bloc se compose de l'identifiant de la page, l'identifiant donné au premier élément `<TextBlock>` et enfin une traduction de ce dernier composant en « blockCount1 » qui commence compter les blocs à partir du numéro 1 au lieu d'à partir de zéro, comme font souvent les logiciels HTR. Pour donner encore un exemple, l'identifiant de la première ligne de texte du premier block sur l'onzième folio porterait l'identifiant « f11-textblock_0-textline_0-lineCount1 ». Encore, l'identifiant se compose des étiquettes des éléments parents (f11, textblock_0) ainsi qu'une traduction du dernier composant (textline_0) en une chaîne plus logique (lineCount1).

1.1.2 La page en particulier

Selon notre modélisation, l'élément TEI `<surface>` représente une page et contient donc toute donnée encodée dans l'élément ALTO `<Page>` et son enfant `<PrintSpace>`. Souvent le fichier ALTO présente le longueur et l'hauteur de la page dans les éléments `<Page>` et `<PrintSpace>`. Cet redondance se produit au niveau de la page parce que l'entièreté de la page traitée est aussi ce qui est transcrit. Quand on construit un fichier ALTO à partir du document TEI il faut recréer cette redondance en répétant le longueur et l'hauteur dans les deux éléments ALTO. La transformation en TEI représente ces deux données dans un seul élément, le `<surface>`. Imbriqué dans l'élément `<surface>`, l'élément `<graphic>` combine l'ARK du fac-similé numérique et l'URI IIIF pour l'IIIF Image API de la BnF. Cet URI renvoie l'image entière de la page numérisée. L'exemple du `<Page>` et l'exemple de sa modélisation en TEI sont donnés dans la Figure 3. Une visualisation de la transformation d'ALTO à TEI est donnée à la fin de ce chapitre, dans la Figure 8.

```
<Layout>
  <Page WIDTH="2568" HEIGHT="3631" PHYSICAL_IMG_NR="2" ID="page_2">
    <PrintSpace HPOS="0" VPOS="0" WIDTH="2568" HEIGHT="3631">
<!-- ... -->
    </PrintSpace>
  </Page>
</Layout>
```

(a) Le `<Page>` en ALTO

```
<surface xml:id="f11" n="2" ulx="0" uly="0" lrx="2568" lry="3631">
  <graphic url="https://gallica.bnf.fr/iiif/ark:/12148/btv1b8610802d/
    f11/full/full/0/native.jpg"/>
<!-- ... -->
</surface>
```

(b) Le `<surface>` en TEI

FIGURE 3 – La modélisation du `<Page>`

1.2 Le bloc

La *SegmOntoZone* indique un bloc sur la page. Elle peut contenir du texte, comme dans le cas d'une *MainZone*, ou elle peut n'en avoir pas, comme dans le cas d'une *GraphicZone* qui décrit la région d'une page dans laquelle se trouve un dessin. Un fichier ALTO encode tout type de bloc dans l'élément `<TextBlock>` même s'il ne contient pas du texte. Étant modélisées en TEI, les données du `<TextBlock>` sont transformées en l'élément `<zone>`.

1.2.1 Des règles générales pour l'élément <zone> dans le modèle TEI

Avant d'aller en plus de détail particulier à la transformation du <TextBlock> en TEI, il faut parler de certaines transformations généralisées pour plusieurs éléments du fichier ALTO. Dans notre modélisation, l'élément TEI <zone> représente plus que le <TextBlock> du schème ALTO. En fait, il représente tout masque prédit par des modèles HTR. La région prédite d'un bloc (*SegmOntoZone*) et celle d'une ligne de texte (*SegmOntoLine*) ainsi que celle d'un mot et celle d'un glyphe sont toutes représentées par l'élément TEI <zone>. Il est bien adapté à représenter les données géométriques d'un masque.

Le <zone> doit porter certains attributs d'usage, peu importe quel type de masque il représente. Ces attributs décrivent l'étiquette (@type) appliquée à la région décrite et les quatre coordonnées (@HPOS, @VPOS, @WIDTH, @HEIGHT) du rectangle qui l'encadre. Comme explique la section **??**, les valeurs des attributs @HPOS et @VPOS font les coordonnées x et y, respectivement, du point le plus haut à gauche du rectangle, comme se voit dans la Figure **??**. La valeur de l'attribut @HEIGHT compte la différence entre le point le plus haut et le point le plus bas du rectangle. La valeur de l'attribut @WIDTH calcule aussi la différence entre le côté gauche du carré et son côté droit. En outre, les quatre coordonnées du rectangle se sont transformés afin de construire l'attribut @source pour tout <zone>. Le @source fournit l'URL pour visionner la région de l'image dans un API IIIF. Selon les normes de l'IIIF, l'URL se compose des parties suivantes :

titre	exemple
scheme	https ://
server	gallica.bnf.fr
prefix	/iiif/ark :/12148
identifier (/ARK/folio)	/btv1b8610802d/f11
nombre de pixels entre la position 0 et la position la plus à gauche de la région sur l'axe des x (@HPOS en ALTO)	323
nombre de pixels entre la position 0 et la position la plus en haute de la région sur l'axe des y (@VPOS en ALTO)	336
nombre de pixels entre la position la plus à gauche et celle la plus à droite sur l'axe des x (@WIDTH en ALTO)	2056
nombre de pixels entre la position la plus en haute et celle la plus en bas sur l'axe des y (@HEIGHT en ALTO)	2812
size	full
rotation	0
quality	native
.format	.jpg

Les composants de la table ci-dessus constituent l'URL suivant :

```
https://gallica.bnf.fr/iiif/ark:/12148/btv1b8610802d/f11
    /323,336,2056,2812/full/0/native.jpg
```

Cet URL se donne comme la valeur de l'attribut `@source` des éléments `<zone>` dans notre modélisation TEI. Il permet de visionner la partie du fac-similé numérique concernée depuis un éditeur, tel que *TEIPublisher*, qui requête l'image de l'API IIIF.

Normalement, les modèles HTR d'aujourd'hui prédisent le rectangle qui encadre la région sur la page et aussi le polygone qui fait un masque plus précis. Si le modèle produit les deux formes de masque, il les font uniquement pour les régions sur la page qui contiennent soit du texte, soit une image. Dit autrement, tout type de région, y compris l'espace entre mots, s'encadre dans un rectangle, mais les types qui contiennent quelque chose autre qu'une espace vide, donc toute région sauf l'espace entre mots, s'encadrent dans un polygone. Le polygone porte plus de coordonnées que le rectangle. Dans le schème ALTO, les valeurs des coordonnées du polygone sont données dans l'attribut `@POINTS` de l'élément `<Polygon>` qui descend indirectement de l'élément sur lequel il porte. Notre modélisation en TEI représent les coordonnées du polygone dans l'attribut `@points` du même élément `<zone>` qui est concerné.

1.2.2 Le bloc (`<TextBlock>`) en particulier

En plus des données d'usage (le type et les coordonnées du polygone et/ou du rectangle) la modélisation TEI du `<TextBlock>` exige la composition d'URL pour visionner le masque du bloc (`@source`) et la décomposition de l'étiquette appliquée au bloc. Entraîné sur le vocabulaire *SegmOnto*, le modèle HTR devrait donner au bloc une référence à une étiquette qui peut se composer de trois parties : le type, le sous-type, et le numéro dans la suite. Le deuxième colonne prédite sur la page, par exemple, porterait l'étiquette *MainZone :column :2*. Les étiquettes ainsi composées sont données aux blocs et aux lignes de texte. Les parties du document encore plus petit, tel que le mot ou le glyphe, ne portent pas d'étiquette composée.

Par conséquent, uniquement les étiquettes attribuées aux éléments ALTO `<TextBlock>` et `<TextLine>` sont décomposées lors de leur transformation TEI. Elles peuvent se diviser en trois. L'attribut `@type` prend la première partie, le `@subtype` prend le sous-type qui pourrait suivre les deux points, et le `@n` prend le numéro s'il y en a un qui suit les deux points à la fin. Si le modèle HTR n'a pas mis en pratique des étiquettes aussi détaillées, les attributs `@subtype` et `@n` prennent la valeur *none* pour le bloc. Mais pour la ligne de texte (`<TextLine>`), la valeur du numéro se constitue du compte que fait l'application `alto2tei` des lignes de texte traitées sur la page. Un tel compte n'est pas si logique pour les blocs et donc l'attribut `@n` ne porte pas de valeur significative si l'étiquette n'en a pas donnée aucune. L'exemple du `<TextBlock>` et l'exemple de sa modélisation en TEI sont donnés dans la Figure 4. Une visualisation de la transformation d'ALTO à TEI est donnée à la fin du chapitre dans la Figure 9.

```
<TextBlock HPOS="323" VPOS="336" WIDTH="2056" HEIGHT="2812" ID="
    textblock_0" TAGREFS="BT2062">
  <Shape>
    <Polygon POINTS="2379 336 2379 3148 323 3148 323 336"/>
  </SHAPE>
<!-- ... -->
</TextBlock>
```

(a) Le `<TextBlock>` en ALTO

```
<zone xml:id="f11-textblock_0-blockCount1" type="MainZone" corresp="#
    MainZone" subtype="none" n="none" ulx="323" uly="336" lrx="2379"
    lry="3148" points="379,336 2379,3148 323,3148 323,336" source="
    https://gallica.bnf.fr/iiif/ark:/12148/btv1b8610802d/f11
    /323,336,2056,2812/full/0/native.jpg">
<!-- ... -->
</zone>
```

(b) Le `<zone>` du `<TextBlock>` en TEI

FIGURE 4 – La modélisation du `<TextBlock>`

1.3 La ligne de texte

Dans le vocabulaire *SegmOnto*, l'étiquette *line* s'applique à la région de l'image dans laquelle s'encadre une ligne de texte. L'élément ALTO qui prend cette donnée est l'élément `<TextLine>`. Contrairement au `<TextBlock>` qui ne contient pas forcement du texte, l élément `<TextLine>` doit avoir des prédictions du texte encodées dedans et doit donc avoir d'enfants qui descendent de lui. Comme l'étiquette *SegmOnto* du `<TextBlock>`, celle du `<TextLine>` se divise en trois parties. Si la valeur du `@type`, la première partie de l'étiquette, est identique à l'une des étiquettes listée dans le `<taxonomy>` du `<teiHeader>`, l'élément portera aussi l'attribut `@corresp` qui prendra comme valeur une référence à la classe.

La modélisation des données du `<TextLine>` en TEI s'appuient comme d'habitude sur l'élément `<zone>` parce qu'il porte sur la représentation d'une région de la page et une partie des données du `<TextLine>` portent sur le masque de la ligne. Les attributs du `<zone>` pour la ligne de texte sont identiques à ceux du bloc de texte. Il y a les quatre coordonnées du rectangle `@ulx`, `@uly`, `@lrx`, `@lry` récupérées respectivement depuis les attributs suivants du `<TextLine>` : `@HPOS`, `@VPOS`, `@WIDTH`, `@HEIGHT`. Ensuite l'attribut `@source` se compose en part de ces quatre coordonnées. Enfin, le `<zone>` contient les points du `<Polygon>`, l'élément qui descend du `<TextLine>` et qui décrit le périmètre du polygone qui encadre la ligne de texte.

Le `<zone>` du `<TextLine>` contient deux enfants directs particulier à la ligne de texte : le `<line>` et le `<path>`. L'élément `<line>` contient le texte de la ligne. Comme attribut, il porte simplement un identifiant et le nombre de la ligne

de texte lors du traitement du fichiers ALTO. Le `<path>` représente le *baseline* de la ligne de texte, c'est-à-dire le début et la fin de la ligne linéaire. Il se compose donc de quatre nombres, un pair x,y indiquant le point du début et un deuxième pair x,y indiquant le point de la fin. Les quatre nombres sont encodés directement dans le fichier ALTO comme la valeur de l'attribut `@BASELINE` du `<TextLine>`. L'élément TEI qui convient le mieux à la donnée du *baseline* est l'élément `<path>`. L'exemple du `<TextLine>` et l'exemple de sa modélisation en TEI sont donnés dans la Figure 5. Une visualisation de la transformation d'ALTO à TEI est donnée dans la Figure 10.

```
<TextLine ID="textline_0" TAGREFS="LT722" BASELINE="605 944 2010 925"
    HPOS="596" VPOS="777" WIDTH="1414" HEIGHT="182">
  <Shape>
    <Polygon POINTS="605 944 596 816 666 795 669 795 672 795 814 810
    838 792 838 789 841 789 844 789 847 789 932 804 953 789 956 789 959
     789 962 789 1050 801 1323 783 1326 783 1704 798 1768 777 1771 777
    1774 777 2004 798 2010 925 2004 953 1798 941 1750 956 1747 956 1744
     956 605 959"/>
  </Shape>
  <String CONTENT="A MONSIEVR" HPOS="596" VPOS="777" WIDTH="1414"
    HEIGHT="182"/>
</TextLine>
```

(a) Le `<TextLine>` en ALTO où le texte est contenu dans l'attribut `@CONTENT` de l'élément descendant `<String>`

```
<zone xml:id="f11-textblock_0-textline_0-lineCount1" type="HeadingLine"
     corresp="#HeadingLine" subtype="none" n="none" ulx="596" uly="777"
     lrx="2010" lry="959" points="605,944 596,816 666,795 669,795
    672,795 814,810 838,792 838,789 841,789 844,789 847,789 932,804
    953,789 956,789 959,789 962,789 1050,801 1323,783 1326,783 1704,798
     1768,777 1771,777 1774,777 2004,798 2010,925 2004,953 1798,941
    1750,956 1747,956 1744,956 605,959" source="https://gallica.bnf.fr/
    iiif/ark:/12148/btv1b8610802d/f11/596,777,1414,182/full/0/native.
    jpg">
  <path xml:id="f11-textblock_0-textline_0-lineCount1-baseline" points=
    "605,944 2010,925"/>
  <line xml:id="f11-textblock_0-textline_0-lineCount1-text" n="1">A
    MONSIEVR</line>
</zone>
```

(b) Le `<zone>` du `<TextLine>` en TEI

FIGURE 5 – La modélisation du `<TextLine>`

1.3.1 La ligne de texte quand il y a des prédictions sur les mots et les glyphes dedans

Pour certains fichiers ALTO, tel que ceux qui sortent de l'interface *eScriptorium*, la modélisation en TEI s'arrête là. Le fichier ALTO ne porte pas de

plus de détail après la ligne de texte. Mais pour certains d'autres fichiers, tel que ceux qui sont produits par l'engin *Kraken* depuis la ligne de commande, ils attestent aux prédictions sur des mots et sur des glyphes. Dans ce cas, la ligne de texte a plus de descendants, mais afin de garder une arborescence générique qui supporte des comparaisons entre des fichiers de divers formats, le `<zone>` de la ligne de texte garde toujours les mêmes deux enfants : le `<path>` et le `<line>`. En plus de ces deux, le `<zone>` contient une suite d'éléments `<zone>` pour tout segment prédit sur la ligne, soit un mot, soit une espace entre mots.

1.4 Le segment

Les fichiers ALTO qui contiennent plus de détail ont des éléments `<zone>` pour des segments (`<String>`) prédits et pour les glyphes (`<Glyph>`) prédits dans les segment. Les modèles HTR prédisent les segments qui contiennent des glyphes, tel qu'un mot, et ceux qui n'en ont pas, tel qu'une espace entre mots. Les segment qui contiennent des glyphes peuvent représenter la prédiction d'un mot, de la ponctuation, ou d'un mot avec de la ponctuation à côté. L'important est que le segment contient soit la prédiction d'au moins un glyphe ou la prédiction d'une espace entre mots. Uniquement les segments (`<String>`) qui contiennent des prédictions sur des glyphes portent aussi un polygone dans lequel s'encadre la chaîne des glyphes. Si le `<String>` représente une espace entre mots, le masque prédit n'est qu'un rectangle. Bien que l'élément `<String>` ne porte pas directement sur le texte prédit, le modèle HTR évalue son taux de réussite à partir des glyphes bien prédits dedans. L'évaluation de sa prédiction est représentée dans l'attribut `@WC` de l'élément `<String>`. L'acronyme *WC* signifie en anglais *word confidence*. L'exemple du `<String>` et l'exemple de sa modélisation en TEI sont donnés dans la Figure 6. Une visualisation de la transformation d'ALTO à TEI est donnée dans la Figure 11.

```
<String ID="segment_1" CONTENT="MONSIEVR" HPOS="837" VPOS="777" WIDTH="
    1172" HEIGHT="182" WC="0.9.64">
  <Shape>
    <Polygon POINT="..."/>
  </Shape>
<!-- ... -->
</String>
```

(a) Le `<String>` en ALTO où le texte n'est pas contenu dans l'attribut `@CONTENT` de l'élément descendant `<String>`

```
<zone xml:id="f11-textblock_0-textline_0-segment_2-segCount3" type="
    String" ulx="837" uly="777" lrx="2009" lry="959" points="..."
    source="https://gallica.bnf.fr/iiif/ark:/12148/btv1b8610802d/f11
    /837,777,1172,182/full/0/native.jpg">
  <certainty xml:id="f11-textblock_0-textline_0-segment_2-segCount3-
    cert" target="#f11-textblock_0-textline_0-segment_2-segCount3-text"
     locus="value" degree="0.9064"/>
<!-- ... -->
</zone>
```

(b) Le `<zone>` du `<String>` en TEI

FIGURE 6 – La modélisation du `<String>`

1.5 Le glyphe

Les caractères et de la ponctuation prédits par des modèles HTR sont tous encodés dans l'élément `<Glyph>` selon le schème ALTO. Contrairement au `<String>` qui sert à contenir une chaîne de glyphes, le `<Glyph>` du fichiers ALTO contient à la fois le masque et le texte. Pour cette raison, la modélisation en TEI représente, comme d'habitude, le masque du `<Glyph>` dans l'élément `<zone>` et le texte prédit dans l'élément `<c>`. Ce dernier est un élément du schème TEI destiné à la représentation d'un caractère, soit une lettre, soit de la ponctuation. Il convient bien donc à la représentation de toute prédiction dans l'élément `<Glyph>` du fichier ALTO. Le modèle HTR évalue son taux de réussite de sa prédiction du glyphe. L'évaluation de sa prédiction est représentée dans l'attribut `@GC` de l'élément `<String>`. L'acronyme *GC* signifie en anglais *glyph confidence*. L'exemple du `<Glyph>` et l'exemple de sa modélisation en TEI sont donnés dans la Figure 7. Une visualisation de la transformation d'ALTO à TEI est donnée dans la Figure 12.

```
<Glyph ID="char_1" CONTENT="M" HPOS="837" VPOS="777" WIDTH="159" HEIGHT
    ="162" WC="0.8127">
  <Shape>
    <Polygon POINT="..."/>
  </Shape>
<!-- ... -->
</String>
```

(a) Le `<Glyph>` en ALTO

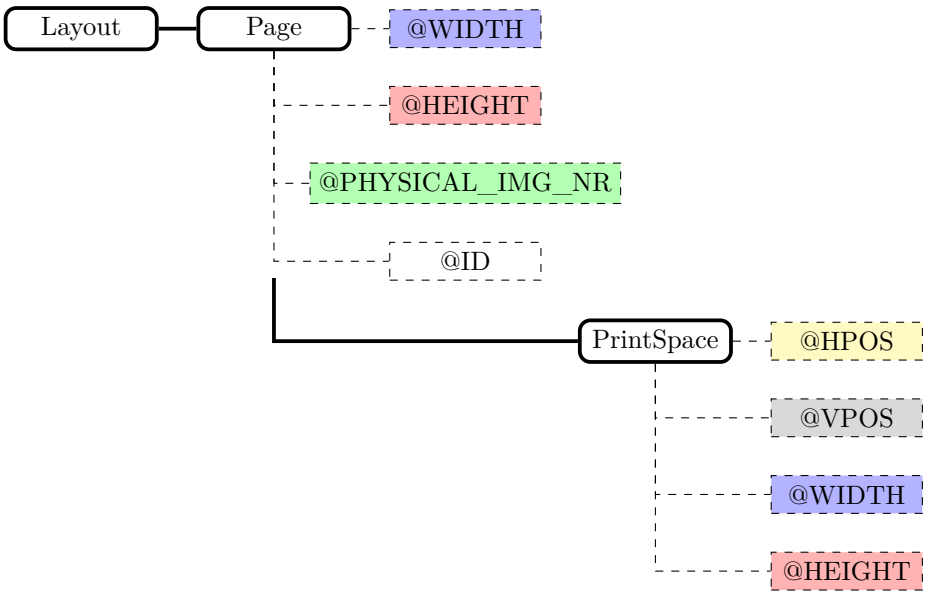
```xml
<zone xml:id="f11-textblock_0-textline_0-segment_2-char_1-glyphCount2"
    type="String" ulx="837" uly="777" lrx="996" lry="939" points="..."
    source="https://gallica.bnf.fr/iiif/ark:/12148/btv1b8610802d/f11
    /837,777,154,120/full/0/native.jpg">
  <certainty xml:id="f11-textblock_0-textline_0-segment_2-char_1-
    glyphCount2-cert" target="#f11-textblock_0-textline_0-segment_2-
    char_1-glyphCount2-text" locus="value" degree="0.8127"/>
  <c xml:id="f11-textblock_0-textline_0-segment_2-char_1-glyphCount2-
    text">M</c>
</zone>
```

(b) Le `<zone>` du `<Glyph>` en TEI

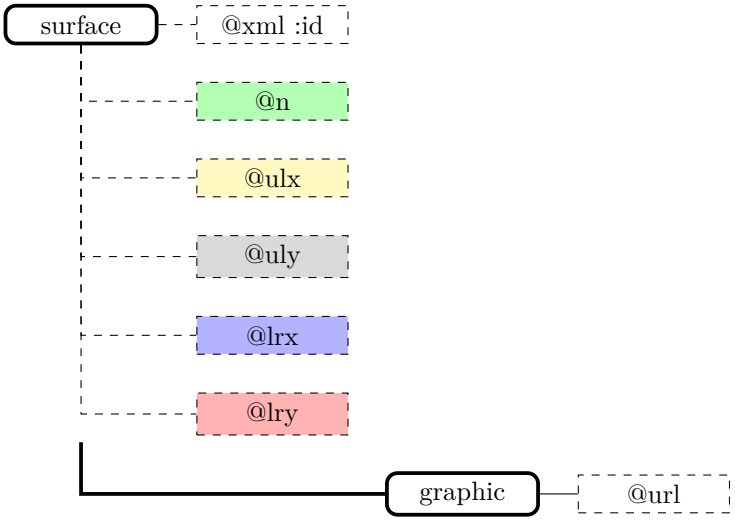
FIGURE 7 – La modélisation du `<Glphy>`

2 Les visualisations de la transformation

Des visualisations de la modélisation de chaque élément du fichier ALTO sont montrées dans les figures qui suivent. Les attributs sont visualisés par les carrés en ligne tirée et les éléments XML sont visualisés par les carrés en ligne solide. La couleur de l'élément ou de l'attribut du schème ALTO est répétée dans la visualisation du schème TEI quand sa valeur est utilisée.

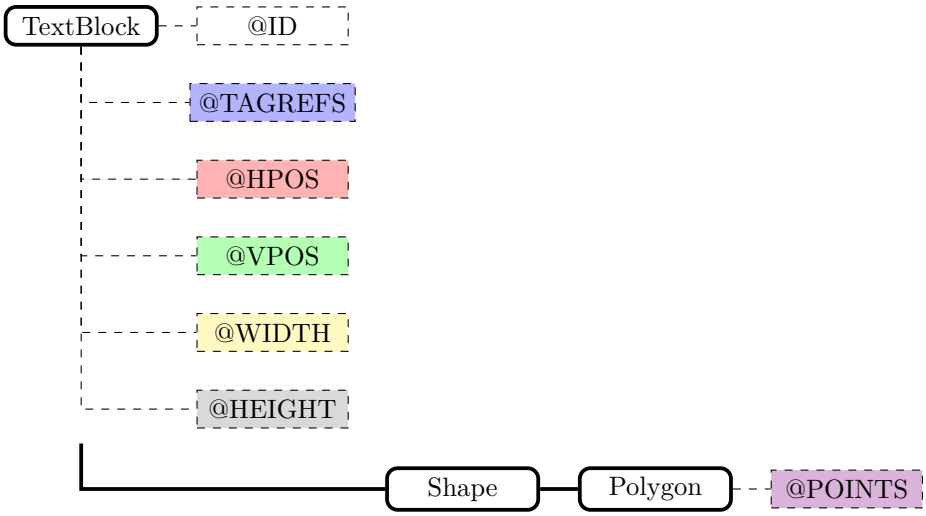


(a) Le modèle du `<Page>` en ALTO

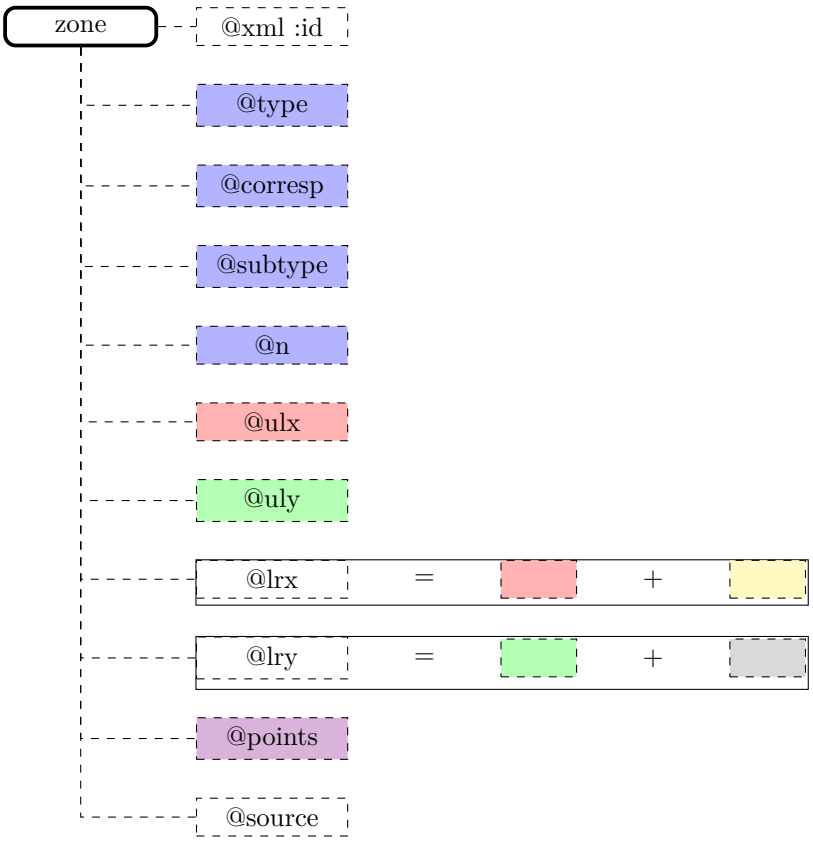


(b) La modélisation du `<Page>` en TEI

FIGURE 8 – La transformation du `<Page>` d'ALTO à TEI

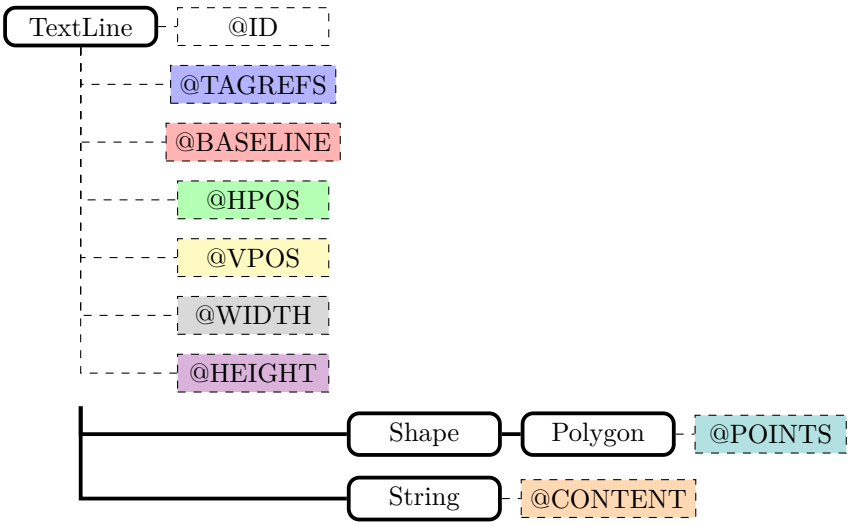


(a) Le `<TextBlock>` en ALTO

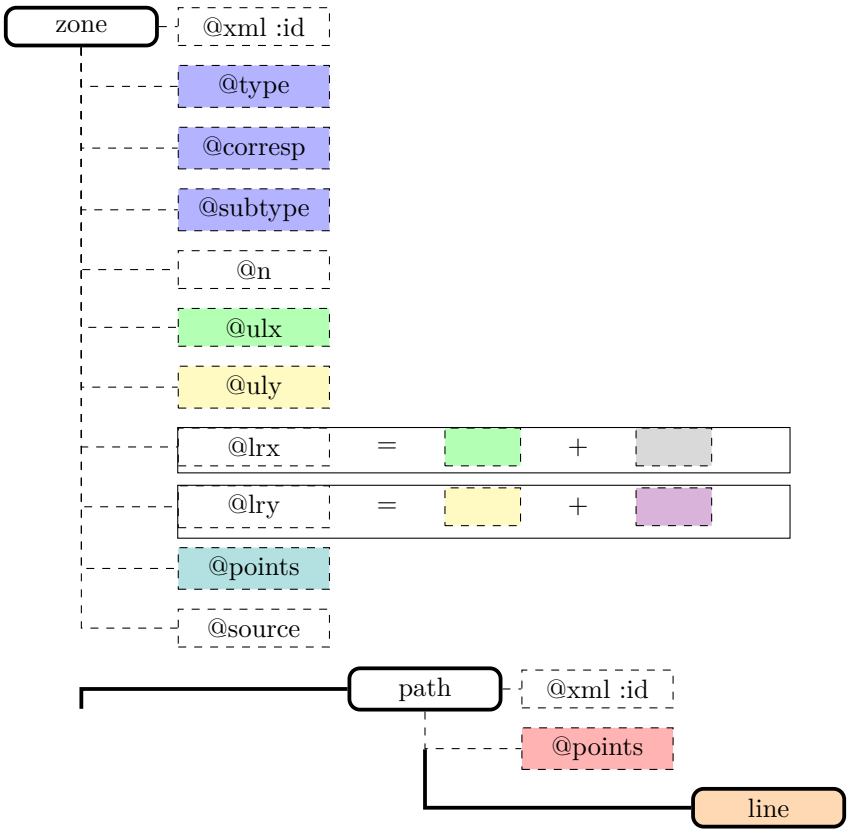


(b) La modélisation du `<TextBlock>` en TEI

FIGURE 9 – La transformation du `<TextBlock>` en TEI



(a) Le `<TextLine>` en ALTO où tout le contenu textuel prédit de la ligne est présenté dans l'attribut `@CONTENT`



(b) La modélisation du `<TextLine>` en TEI

FIGURE 10 – La transformation du `<TextLine>` en TEI

(a) Le `<String>` en ALTO

(b) La modélisation du `<String>` en TEI

FIGURE 11 – La transformation du `<String>` en TEI

(a) Le `<Glyph>` en ALTO

(b) La modélisation du `<Glyph>` en TEI

FIGURE 12 – La transformation du `<Glyph>` en TEI