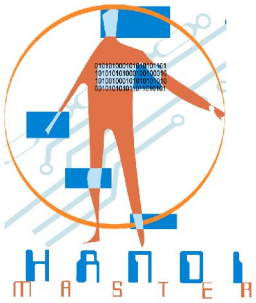
PARIS 8
UNIVERSITE
VINCENNES - SAINT-DENIS



MASTER HANDI

Technologies et Handicaps

Projet Technique

NAT

« Un transcripteur universel de documents standards en texte braille. »

- Modélisation et conception du logiciel
- Développement des modules de transcription en braille français intégral, abrégé et mathématique

Présenté par : **Bruno BLANCHARD et Bruno MASCRET**

Directeur du projet : **J. LOPEZ KRAHE**

Saint-Denis, Février 2006

Remerciements

Nous dédions ce modeste travail à :

Tous nos amis, à qui nous souhaitons beaucoup de bonheur.

Toute la promotion Master Handi 2005-2006.

Nos remerciements s'adressent tout particulièrement à Frédéric Schwebel et ses collaborateurs (projet BraMaNet, Mission Handicap de Lyon 1), Henrik Just (Writer2Latex), la communauté des développeurs d'OpenOffice et Benoît Dasset (fonte BrailleAntoine), pour les éléments de programmation intégrés.

Ils s'adressent également à Jaime Lopez Krahé (Directeur de l'UFR Langages Informatique Technologie de l'Université Paris 8, et responsable du Master Handi) et Pascale Pousset (gestion du projet master Handi), qui ont bien voulu diriger ce travail tout au long de sa réalisation.

Nous remercions aussi Marc Ollier (CNEFEI de Suresnes), pour ses conseils, Hélène Deslandes, Vivian Dumont et Jean Pascal Lautrette (étudiants du Master d'ergonomie Paris 8), pour leur participation à l'étude de l'ergonomie de l'interface, et les collaborateurs de la Mission Handicap de Lyon 1, qui ont accepté d'enrichir notre travail par leur jugement en tant qu'utilisateurs et testeurs de notre réalisation.

Enfin, nos plus vifs remerciements s'adressent aux familles Blanchard et Mascret, Aude Lancelle et Isabelle Blanchard, pour leur infinie patience et leur soutien moral tout au long du projet.

Enfin, un grand remerciement à tous ceux qui nous ont aidé de près ou de loin à achever ce travail.

Sommaire

Introduction

Pour les non voyants, l'informatique offre de nouveaux outils d'accès à l'information, lesquels tendent à remplacer les moyens traditionnels nécessitant l'intervention constante d'intermédiaires humains. Pour utiliser un ordinateur les personnes aveugles ont à leur disposition la synthèse vocale, des plages ou les terminaux brailles qui permettent de lire directement en Braille les textes affichés à l'écran et des imprimantes brailles, les embosseuses.

L'ordinateur couplé avec un terminal braille favorise la lecture, pour le non-voyant, de textes littéraires ou d'écritures " en ligne " mais, on imagine sans peine les problèmes de compréhension qui vont se poser dans les domaines scientifiques chaque fois qu'un texte comporte des expressions un tant soit peu complexes, comme en ce qui concerne l'écriture mathématique.

Enfin, si le World Wide Web s'impose aujourd'hui comme un standard pour la diffusion d'information, il ne résout pas tous les problèmes d'accessibilité, notamment en ce qui concerne la représentation de formes graphiques non traduisibles.

Notre modeste travail aura donc pour ambition de contribuer à l'amélioration de l'accessibilité et de la diffusion de l'information (scientifique, littéraire, musicale, langues étrangères…) à destination des personnes non-voyantes, en fournissant un système de conversion de documents intégrant pour chaque écriture braille spécifique un module transcripteur indépendant.

Pour des raisons de temps, nous développerons dans un premier temps les modules " braille littéraire français (intégral) ", " braille littéraire français (abrégé) " et " braille mathématique français ".

Le présent rapport de mémoire est organisé de la façon suivante :

- Le premier chapitre définit le cadre théorique de cet outil, par le recensement des besoins des utilisateurs non-voyants et des transcripteurs. Après une étude de l'existant et du contexte, nous définissons l'objectif que nous nous sommes fixé.
- Le chapitre deux est consacré à l'étude d'une technologie populaire de transformation de document : les filtres XSL, le processeur XSLT et le langage XML.
- Le chapitre trois présente l'analyse formalisée de notre solution, et se termine par la justification de nos choix techniques d'implémentation.
- Nous abordons au chapitre quatre le développement des différents modules du système, notre collaboration avec une équipe d'ergonomes pour l'interfaçage du logiciel et finissons par la présentation des tests unitaires.
- Le chapitre cinq est consacré entièrement aux tests utilisateurs.
- Le chapitre six présente le travail de communication réalisé autour de NAT.
- Pour finir, nous proposons une réflexion sur notre travail, les résultats obtenus, et les perspectives d'évolution.

Chapitre 1 : Analyse préalable

Ce premier chapitre présente une étude de l'existant et du contexte, puis l'objectif que nous nous sommes fixé.

1.1 Présentation du contexte et des problématiques

1.1.1 Introduction et premier aperçu du braille

Le braille, du nom de son inventeur le français Louis Braille, est un langage fortement lié à la langue qu'il décrit, parfois même au pays dans lequel il est utilisé. Il existe un braille français, belge, allemand, anglais et américain...

Chaque sous-langage est donc fondamentalement différent d'une langue à une autre (l'abrégé français n'a rien avoir avec l'abrégé anglais) et même d'un pays à un autre (les brailles mathématiques français et québécois sont différents).

Seule la notation musicale conserve son universalité.

Le braille est un langage vivant, sujet à de fréquentes évolutions. La tendance actuelle est logiquement à l'unification des différents brailles décrivant la même langue, mais ce travail reste lent et compliqué, comme toute entreprise normative.

La plupart des pays disposent de structures publiques ou privées chargées du suivi de ces évolutions (commissions ministérielles, associations, centre de transcription...). Certaines ne sont parfois même pas reconnues officiellement, bien que faisant autorité dans le pays.

Une des premières difficultés va donc venir de ce caractère hétérogène et complexe. Nous détaillerons dans ce qui suit d'autres problèmes liées à la forme de ce langage.

1.1.2 Les besoins des utilisateurs non-voyants et des transcripteurs

Une grande partie des besoins des utilisateurs non-voyants est liée à la transcription en braille de documents de différentes natures, de l'impression et de la lecture de ses transcriptions que l'on pourrait lister comme suit :

- Disposer d'un outil unique de transcription en braille gérant l'ensemble des traitements à effectuer;
- Disposer d'un outil de transcription gérant les différentes écritures brailles, dans différentes langues;
- Disposer d'un grand nombre de formats d'entrée;
- Pouvoir imprimer en braille les transcriptions (à domicile ou dans un centre de transcription);
- Pouvoir lire directement les transcriptions sur une plage braille;
- Utiliser une interface homme machine ergonomique permettant à la fois une utilisation et un paramétrage simple.

1.1.3 Un exemple concret : L'accès aux mathématiques pour les non-voyants

Les problèmes rencontrés pour la transcription en braille de textes mathématiques concernent en grande majorité des documents produits par des professeurs de mathématiques.

Ces documents sont très souvent difficiles à exporter sur d'autres machines ou à utiliser avec d'autres logiciels, voire avec une version différente du même logiciel. En effet, il existe toujours plusieurs manières de produire un document électronique et d'obtenir un " effet souhaité " à l'écran ou à l'impression. Mais ensuite, lorsque le fichier est envoyé à un collègue, à une revue, ou simplement consulté sur un autre ordinateur, il s'avère qu'il n'est pas " lisible " par les interlocuteurs.

Les enjeux de l'enseignement pour les élèves handicapés sont importants. Il faut d'abord éviter l'exclusion et ensuite favoriser l'intégration de ces élèves. Le pourcentage d'aveugles qui poursuivent des études scientifiques est dérisoire alors que des perspectives d'insertion professionnelles dans ce domaine sont tout à fait réelles.

Lire et écrire les mathématiques est fondamentalement différent de lire et écrire du texte. Si le braille est adéquat en ce qui concerne la représentation d'un texte, il en va tout autrement pour les mathématiques. Arthur I. KARSHMER de l'University of South Florida, USA, à résumé, lors de la Journée thématique BrailleNet 2002 [WEB09], deux raisons simples à cela : la linéarité et le jeu de caractères.

1.1.2.1 La linéarité :

Le texte est de nature linéaire, alors que les équations mathématiques sont bi-dimensionnelles. Ce que vous êtes en train de lire dans ce texte constitue un bon exemple de ce problème. En revanche, examinez l'équation relativement simple qui suit :

$$a = \sqrt{\frac{x^2 - y}{z}}$$

On remarquera immédiatement que l'équation contient un exposant et une fraction - les deux étant de nature bi-dimensionnelle. L'équation pourrait avoir été écrite de manière linéaire, par exemple: a = rac(((x exp 2) - y) / z)

Pour cette équation relativement simple, une représentation linéaire est adéquate afin qu'un utilisateur non-voyant puisse la lire. Avec une écriture plus complexe de l'équation, il est vraisemblable que la représentation linéaire perde toute son utilité.

1.1.2.2 Le jeu de caractères :

Le texte peut être généralement représenté dans un nombre limité de caractères qui incluent normalement les lettres majuscules et minuscules, les 10 chiffres, plusieurs marques de ponctuation, et un petit jeu de caractères spéciaux. D'autre part, les équations peuvent contenir tous les caractères pouvant figurer dans des textes normaux, ainsi qu'un grand nombre de caractères spéciaux.

Le braille normal 6 points peut intrinsèquement seulement représenter 64 caractères (2^6). Par l'utilisation de séquences spéciales d'échappement, le braille peut supporter un jeu de caractères beaucoup plus grand. Mais ceci a un prix : Les caractères basiques qui peuvent être représentés en Braille peuvent avoir différentes significations selon le contexte.

S'il n'y a effectivement pas de limitation du nombre de caractères qu'on peut représenter, la lecture et la compréhension de ces séquences spéciales deviennent très complexe. Plus nous avons besoin de caractères nouveaux, plus il nous faut connaître des séquences. Par exemple, la lettre "a" peut avoir plusieurs significations selon le contexte. Cela pourrait être un "a", un "A", un "α", et ainsi de suite. Ceci rend la lecture et/ou l'écriture en braille des équations mathématiques difficiles.

Ajoutez maintenant la nature multidimensionnelle des équations, et même des plus simples! Il devient très clair que la représentation des mathématiques en braille est difficile compte tenu du jeu de caractères qui peut être représenté en braille 6 points.

Malgré ces limites, de nombreuses notations mathématiques Braille ont été développées au cours des dernières années, principalement basées sur le braille 6 points.

1.2 Les transcripteurs braille existants

Il existe un certain nombre d'outil de transcription en braille, fonctionnant de manière indépendante :

1.2.1 Braille littéraire :

WinBraille et Duxburry gèrent le braille intégral et l'abrégé.

Winbraille utilise des fichiers de règles de transcription (rule files), qui ne sont pas toujours à jour ou exhaustives.

Duxburry est un logiciel payant assez onéreux pour une personne voyante. Ses évolutions sont laborieuses le logiciel n'est pas libre et développé aux États-Unis, les spécialistes français chargés de son adaptation ne disposent que d'une partie des sources du logiciel.

1.2.2 Braille mathématique :

BraMaNet [Web07] est un traducteur d'expressions mathématiques en braille. Il fonctionne de manière optimale avec le MathML généré par MathType (extension payante de l'éditeur d'équations de Word) et un peu moins bien avec le MathMl d'OpenOffice. Il devrait prendre en compte le LaTeX d'ici mi-2006.

L'université de Jussieu travaille aussi sur un traducteur de mathématiques [Web08], plus ambitieux car avec plus de formats possibles d'entrée et de sorties.

1.2.3 Braille musical :

Braille Musical Editor (BME) est comme son nom l'indique un éditeur de musique en braille musical, permettant la lecture et l'impression de documents. C'est a priori le seul outil performant dans son domaine.

1.2.4 Braille chimie, autres brailles (scientifiques, langues étrangères...) :

Il existe à l'heure actuelle d'autres brailles en cours de spécification (braille chimie). Les standards n'ayant pas été arrêtés, il est difficile de réaliser quoi que ce soit pour l'instant.

De plus, il n'y a pas de manière simple d'indiquer la langue dans laquelle un mot est écrit.

1.3 Les techniques développées

Les outils de présentation d'un document électronique à un non voyant ne transcrivaient malheureusement que des chaînes de texte pur, ce qui a exclu jusqu'ici le non voyant de l'accessibilité aux formules mathématiques. Plusieurs techniques ont été développées pour aider les étudiants déficients visuels à se servir des mathématiques :

1.3.1 Représentations tactiles.

Dans une volonté d'éviter le plus possible aux professeurs d'avoir à apprendre le braille, il y a eu des projets leur permettant de préparer leurs documents pédagogiques grâce à des moyens électroniques qui leur sont familiers et de les avoir ensuite automatiquement transcrits en braille.

En utilisant des langages à marquage standard, ce projet permet aux professeurs d'accéder à des outils puissants leur permettant de créer des documents pédagogiques, et de les rendre accessibles aux étudiants non-voyants.

1.3.2 Assistances sonores

L'audio, pour les étudiants non-voyants, a été l'un des médias les plus populaires, l'un de ceux qui ont eu le plus de succès et est utilisé dans une grande variété d'interfaces informatiques.

Les lecteurs d'écran dotés de synthèses vocales (Jaws pour Windows par exemple) sont devenus des outils nécessaires pour les non-voyants utilisateurs d'un ordinateur. Mais ces outils sont souvent incapables de s'adapter à des interfaces plus techniques ou plus spécifiques que celles des logiciels standards.

1.3.3 Représentation sonore

Lire les équations ne constitue pas le seul problème rencontré par les étudiants aveugles en mathématiques. La production d'une fonction, par exemple, peut aussi représenter une structure multidimensionnelle. Ces formes de structure mathématique sont même plus difficiles à décrire en mots qu'avec une équation.

Les représentations strictement tonales ont été testées avec un succès limité surtout dans le cas de graphiques compliqués. L'écoute seule ne suffit pas à produire l'information capable de décrire des objets compliqués.

1.3.4 Périphériques haptiques

Le développement d'afficheur braille de très bonne "résolution" serait la meilleure manière de représenter à la fois du texte et des informations d'autre nature. Malheureusement ces outils sont encore loin d'être à un prix abordable.

D'autres aides, telles que les imprimantes Tiger, peuvent produire des informations en braille papier relativement bon marché, mais elles ne sont pas dynamiques par nature une fois qu'elles ont été embossées. Une autre solution à ce problème pourrait résider dans des outils tactiles ou à réaction artificielle. De tels dispositifs peuvent travailler en deux ou trois dimensions, et sous contrôle informatique.

1.3.5 Approches intégrées

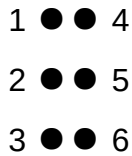
Bien que toutes les approches pour fournir l'enseignement des mathématiques aux étudiants aveugles aient de grands mérites, les plus couronnées de succès présentent plusieurs interfaces techniques.

Dans le domaine de la lecture des équations, par exemple, les projets MATHS et MAVIS utilisent plusieurs medias de production pour accomplir leurs tâches. Ceux-ci comprennent la parole, le son, le braille et des techniques structurées de navigation. Dans le cas de MAVIS, la recherche à partir d'expériences de lecture des mathématiques a préparé le terrain pour une lecture moins ambiguë d'équation et la possibilité pour l'utilisateur du système de dire des équations.

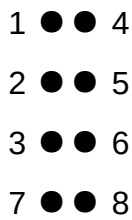
Un autre exemple d'approche intégrée réside dans la combinaison de dispositifs tactiles et sonores pour décrire une donnée graphique. Dans ces projets, des outils de réaction artificielle sont couplés avec des techniques de mise en son pour délivrer un rendu de haute qualité du graphique.

1.4 Normes Braille

Rappelons que le système braille de base est constitué d'une cellule de deux rangées verticales comportant chacune trois points, nommés par convention (de haut en bas) : points 1, 2 et 3 pour la colonne de gauche, points 4, 5 et 6 pour la colonne de droite.



Les caractères braille sont formés par des combinaisons de ces points qui peuvent être soit présents, soit absents. Un tel système permet par conséquent d'obtenir 64 signes (espace compris). En informatique, cependant, les 64 combinaisons ne peuvent suffire à représenter l'ensemble des signes susceptibles d'être affichés sur un écran d'ordinateur (au minimum 256), dans un document de type "texte". C'est pourquoi ont été ajoutés deux points supplémentaires à la cellule braille de base, nommés "point 7" (situé sous le point 3) et "point 8" (situé sous le point 6).



1.4.1 Tables brailles

Une table braille est un tableau associant un caractère en noir à une signification en braille.

Il existe de nombreuses tables, normées ou non. Nous vous proposons d'en étudier une plus particulièrement.

1.4.1.1 La table CBFR1252

En 1992, afin d'éviter une prolifération des tables braille, et pour répondre à un besoin croissant d'un code braille informatique de référence, la Commission pour l'Évolution du Braille Français (CEBF) a décidé la création d'une sous-commission informatique chargée d'élaborer une table braille à 8 points adaptée au code informatique (cf Annexe B).

Malheureusement, le jeu de caractères retenu par la commission est directement lié au système d'exploitation de Microsoft : la table se rend donc dépendante d'une solution technique (qu'elle encourage de ce fait...) et perd l'universalité à laquelle elle aurait pu prétendre en utilisant un jeu de caractères universel (par exemple : UTF8).

Sous Windows, la norme "Unicode" désigne un ensemble de jeux de caractères à 16 bits utilisés pour représenter tous les caractères des langues en vigueur. La norme ISO-8859 a produit différents standards permettant de représenter les caractères appartenant aux langues latines et européennes sur 8 bits (256 caractères).

Le jeu de caractères ISO 8859-1, dénommé aussi Latin-1, correspond à la langue française. Mais la table réellement utilisée sous Windows et ses applications est désignée par Microsoft sous le terme de CP-1252 (ou Win Latin-1), qui correspond à l'ISO 8859-1, à l'exception des codes (décimaux) de 128 à 159. C'est cette table CP-1252, largement répandue sous Windows, qui a servi de base aux travaux de la sous-commission.

La table braille a été nommée de la manière suivante :

- lorsqu'il s'agit de désigner la table braille en tant que norme ou standard, on doit l'appeler " CBFR1252 ", ce qui signifie " Code braille Français pour la table CP- 1252 ";
- lorsqu'il s'agit de désigner la table en tant que jeu de caractères mis à disposition d'un utilisateur de matériel adapté (plage tactile ou embosseuse) on utilise le nom " FRANCAIS (CP-1252) ".

1.4.1.2 Code "imprimeur"

Il s'agit de pouvoir représenter en braille 6 points "embossé" la présence d'un point 7, d'un point 8 ou des deux simultanément. Les trois préfixes retenus sont : point 4, point 5, points 4-6. La combinaison de points 4-6 (signe de majuscule) est tout indiquée pour signifier la présence du point 7.Le point 4 est retenu pour représenter le point 8. Le point 5 est retenu pour signifier : présence des points 7 et 8. En braille embossé à six points, la fréquence de ces trois préfixes est hautement probable.

1.4.1.3 Règles pour la construction de la table CBFR1252

Les numéros des caractères de la table correspondent à leur valeur décimale (de 0 à 255). D'une manière générale, le point 7 indique la majuscule d'un caractère, le point 8 (auquel s'ajoute parfois le point 7 lorsque la combinaison n'est pas disponible) signale fréquemment une "autre alternative" au signe de base en braille 6 points. Le tableau des 256 caractères du CBFR1252 est présenté en annexe A.

1.4.2 La notation Mathématique Braille

Afin de faciliter l'intégration scolaire en milieu et professionnelle des personnes aveugles et de permettre une meilleure transcription automatique par ordinateur, la Commission pour l'Évolution du Braille Français (CEBF) a modifié en 1992 et 1999 la signification de certains caractères du système braille.

Dans ce même esprit et dans un souci de cohérence, cette commission se devait de faire évoluer la codification mathématique braille. Depuis la rentrée scolaire de septembre 2001-2002 est entrée en vigueur une réforme de la notation braille mathématique applicable à l'ensemble des établissements scolaires spécialisés, des centres de formation professionnelle et des producteurs de braille.

Le document " *NOTATION MATHEMATIQUE BRAILLE Mise à jour de la notation mathématique en braille de 1971 (jusqu'au niveau baccalauréat inclus) "* introduit une nouvelle notation (les codes de blocs) permettant de distinguer en braille les parenthèses dues aux écritures mathématiques des parenthèses rajoutées pour linéariser.

1.4.3 Les brailles littéraires

Chaque pays ou chaque langue dispose de deux types d'écriture littéraire ; l'intégral et l'abrégé.

1.4.3.1 Le braille intégral

Le braille intégral est un transcodage de caractère par caractère. C'est la forme la plus simple du braille littéraire.

Le jeu de caractères disponibles n'étant pas suffisant en braille pour permettre une correspondance bijective avec l'écriture " en noir ", il utilise un certain nombre de clefs qui changent l'interprétation des caractères.

Le braille intégral utilise essentiellement deux clefs : la clef de majuscule (points 46 ⠠) et la clef numérique (point 6 ⠼).

Par exemple :

" merci " s'écrira ⠑⠗⠉⠊ ;

" Merci " s'écrira ⠠⠑⠗⠉⠊ (on utilise une fois la clef majuscule ⠠ pour le M);

" MERCI " s'écrira ⠠⠠⠑⠗⠉⠊ (on utilise deux fois la clef pour le mot entier);

" 81 " s'écrira ⠼⠓⠁ (on utilise la clef numérique ⠼).

1.4.3.2 Le braille abrégé

Le braille abrégé est une extension du braille intégral. Aux règles déjà présentes s'ajoutent des règles d'abréviation.

Le braille abrégé est beaucoup plus compliqué que le braille intégral, car il contient de nombreuses exceptions, des règles complexes et nécessite un apprentissage beaucoup plus long (voir le chapitre 4 pour plus de détails).

Il permet cependant à un bon brailliste de lire pratiquement deux fois plus rapidement un texte abrégé qu'un texte intégral.

Le braille abrégé est normé et son utilisation encouragée.

1.5 Langages de marquage et formalismes scientifiques

Sur les réseaux étendus, trois langages de marquage principaux portent l'information scientifique : HTML, MathML et LaTeX.

1.5.1 HTML, MathML et LaTeX

HTML (Hyper Text Markup Language) [WEB03] un format d'échange de fichiers sur le Web, issu du métalangage de description SGML. Il ne contient de définition mathématique que dans ses versions les plus récentes, lesquelles sont peu utilisées dans cet objectif.

Le métalangage XML (Extensible Markup Language) [WEB04], amené à remplacer SGML, possède une définition de type de données (DTD) mathématique appelée MathML (Mathematical Markup Language) [WEB05]. Il existe encore peu de documents écrits dans ce format émergeant.

Enfin, LaTeX est un format plus ancien (associé au traitement de textes du même nom) . De nombreux documents sont écrit sous ce format : c'est une norme de fait pour les documents mathématiques. LaTeX est un support immuable, contrairement à HTML et MathML, lesquels sont dédiés à un support réseau en constante évolution.

Outre l'intégration du format LaTeX dans les versions nouvelles de l'éditeur d'équations de Microsoft Word Mathtype, il existe plusieurs traducteurs vers LaTeX à partir des deux autres formats et réciproquement.

Le langage LaTeX comprend deux types de fichier de données : TeX et DVI. Les fichiers d'extension TeX présentent quelques problèmes d'analyse et de traduction puisqu'on peut y trouver des macro-commandes dont l'usage rend ardue l'analyse syntaxique du document, ainsi que l'analyse lexicale (du fait d'extensions possibles du langage natif). DVI (DeVice Independent) est le format de sortie de LaTeX. Ce type de document (généré par une simple ligne de commande et conçu pour l'impression) ne comprend pas de macros.

Ces trois formats sont tous normés, même si leurs implémentations manquent parfois de rigueur.

1.5.2 Les apports du W3C et les travaux du WAI

Encoder un texte pour qu'il soit visualisable sur de nombreux médias est un problème qui est donc en cours de résolution par un certain nombre de groupes internationaux.

Le W3C (World Wide Web Consortium) [WEB01] avec la norme HTML4, XML et XHTML, apporte une codification des textes pour des supports autres que le simple navigateur Web. Différents types de médias sont d'ores et déjà pris en compte, l'impression papier, le rétroprojecteur, la synthèse vocale et le matériel spécifique Braille (embosseuse, plage Braille). De plus en plus, HTML permet d'introduire des notions sémantiques structurantes du texte et non plus seulement des notions de présentation (sémantique des différents éléments constituant un tableau, niveaux de titre, listes...). Avec XML et XHTML, il est possible de combiner complètement la représentation finale du texte avec sa structure sémantique, c'est-à-dire associer à chaque bloc de textes une correspondance entre le contenu et la forme.

Avec les travaux du WAI (Web Accessibility Initiative) [WEB02], la présentation en braille des documents est donc théoriquement possible. Cependant, seule une minorité de sites appliquent correctement ces recommandations.

L'émergence de la nouvelle recommandation sur XSL (The Extensible Stylesheet Language Family) [WEB06] devrait ouvrir en grand les portes à XML, en fournissant un outil de spécification de réécriture de documents XML, sans avoir à fournir soi-même le moteur de réécriture ; des feuilles de styles XSL sont d'ailleurs d'ores et déjà disponibles pour la conversion en braille.

1.6 Les techniques de transformations de structures

Les techniques de transformation de structures sont largement répandues dans les chaînes de traitement documentaire. On peut identifier trois familles de techniques de transformation fondées sur des principes différents : les filtres, les systèmes de transformation explicite et les systèmes de transformation automatique.

1.6.1 Les filtres

Ils permettent la traduction de documents d'un format vers un autre, les deux formats étant définis à l'avance. Les filtres sont utilisés pour traduire des documents entiers et sont indépendants des autres applications. Ils permettent la transformation des documents appartenant à un format e général unique et produisent un document d'un format préalablement défini.

Ils sont en particulier utilisés pour l'importation et la sauvegarde de documents codés dans un format spécifique à une application. On peut citer les filtres de traduction de LaTeX vers HTML, de document formatés vers des documents SGML, conformes à des DTD particulières comme les DTD d'acquisition (TEI).

1.6.2 Les systèmes de transformation explicite

Ils se basent sur un ensemble de règles guidant la transformation. Ces règles permettent de paramétrer les transformations et leur peuvent, à la différence des filtres, s'appliquer à différents formats de documents.

Cette technique peut être mise en oeuvre par des applications dédiées aux transformations ou intégrées dans d'autres applications (éditeurs de documents, bases de données). Les langages de transformation utilisés dans ces systèmes peuvent être de nature déclarative, comme XSL, impérative comme le langage de balise ou par requêtes comme SgmlQL.

1.6.3 Les systèmes de transformation automatique

Pour effectuer la comparaison entre un format source et un format cible, l'ensemble des règles décrivant chaque format est représenté sous forme d'un arbre (appelé arbre de types).

Pour des raisons d'efficacité, la comparaison est réalisée sur des formes linéaires des arbres de types appelées empreintes. Le résultat d'une comparaison est un ensemble de relations entre un type source et un type cible appelé couplage. La génération d'un document transformé peut alors être effectuée à l'aide de ce couplage.

Leur atout principal vient de leur intégration dans un processus qui peut prendre en compte toutes les caractéristiques des documents (on peut associer des propriétés de style spatial et temporel aux structures transformées). L'analyse des marquages initiaux du document (le marquage " de structure " (titre, chapitres, paragraphes…) et le marquage " syntaxique " qui permet la navigation au sein d'un bloc de texte ou à l'intérieur d'une formule mathématique) est souvent recommandé pour une navigation efficace.

1.7 Nos objectifs

Dans le temps imparti par le projet technique du Master Handi, nous nous proposons :

1. de modéliser de manière exhaustive un logiciel répondant aux besoins des utilisateurs (prévoir de grandes possibilités d'évolution, qui permettront par la suite, i.e. après le projet, d'intégrer d'autres formats d'entrée, de brailles et de sorties);
2. de fournir aux différents utilisateurs un logiciel gratuit de transcription en braille français de documents standards contenant des écritures mathématiques en braille intégral ou abrégé;
3. de gérer plusieurs formats d'entrée, en privilégiant les formats ouverts;
4. d'adapter le logiciel en fonction des niveaux d'expertises et des configurations matérielles des différents types d'utilisateurs (transcripteurs ou non spécialistes, deficients-visuels ou non...);
5. de réaliser un outil indépendant des environnements d'exécution afin de ne pas imposer de configurations spécifiques à l'utilisateur.

Ce premier travail doit être vu comme un point de départ et non pas une fin en soi. Il posera les bases d'un logiciel amené à être complété et amélioré par la suite.

Chapitre 2 : XML et le processeur XSLT

L'objectif de ce chapitre est d 'étudier en détail un processus de transcodage performant, le processeur XSLT, afin de l'intégrer par la suite à notre logiciel.

Nous nous concentrerons sur son utilisation avec des documents structurés en XML, puisque c'est avec ce format que l'utilisation d'XSLT est la plus appropriée.

2.1 Qu'est-ce que XML ?

XML abréviation de e**X**tensible **M**arkup **L**anguage, est un langage de balisage (markup), comme HTML. Contrairement à HTML, qui présente un jeu limité de balises orientées présentation (titre, paragraphe, image, lien hypertexte, etc.), XML est un méta-langage, qui va permettre d'inventer à volonté de nouvelles balises et attributs pour isoler toutes les informations élémentaires ou agrégats d'informations élémentaires.

L'avantage de XML est qu'il nous donne la possibilité de concevoir un balisage plus significatif décrivant précisément le contenu de l'élément, plutôt que d'utiliser des balises décrivant le moyen d'afficher les données. C'est un langage de description et d'échange de données semi structurées qui permet essentiellement de séparer la description structurelle des données de leur réalisation physique.

Cette possibilité permet le balisage de toutes sortes de données, pour structurer toutes sortes d'application, et pas uniquement des données à afficher sur le Web.

2.2 Blocs fondamentaux d'un document XML

Nous ne présenterons dans cette partie que les principaux blocs XML. Nous invitons le lecteur désireux d'approfondir ses connaissances à consulter les nombreuses documentations en ligne.

2.2.1 Déclaration XML

La déclaration XML est en réalité facultative,mais il est fortement recommandé de l'utiliser afin que l'utilisateur ou l'application réceptrice sache qu'il s'agit d'un document XML et connaisse la version utilisée.

Cette déclaration doit commencer le document. On peut aussi définir le langage utilisé pour écrire les données XML (facultatif).

Exemple :

```
<?xml version= "1.0 " encoding="ISO-8859-1" ?>
```

2.2.2 Elément

Les composants les plus importants des documents XML sont les éléments. Tout document XML doit posséder au moins un élément, dans lequel sont imbriqués tous les autres balisages.

Le contenu d'un élément peut être du texte brut (donnée textuelle) ou un arrangement de texte et d'éléments imbriqués (contenu mixte).

Exemple :

```
<?xml version="1.0"?>
<PROMOTION>Ceci est un exemple simple de document xml</PROMOTION>
```

L'élément de plus haut niveau porte le nom " élément document " ou " élément racine ".

2.2.3 Attributs

Les attributs sont placés dans la balise d'ouverture d'un élément, et sont exprimés sous forme de paire Nom/Valeur. La valeur doit toujours être placée entre apostrophes ou guillemets.

Exemple simple :

```
<?xml version="1.0"?>
    <PROMOTION Annee="2005-2006">
        Ceci est un exemple simple de document xml
    </PROMOTION>
```

Annee est un attribut de PROMOTION valant " 2005-2006 ".

Exemple un peu plus complexe :

```
<?xml version="1.0"?>
    <PROMOTION année="2005-2006">
        Liste des étudiants
        <ETUDIANT>
            <NOM>Blanchard</NOM>
            <PRENOM>Bruno</PRENOM>
            <url>brblanchard@wanadoo.fr</url>
        </ETUDIANT>
        <ETUDIANT>
            <NOM>Mascret</NOM>
            <PRENOM>Bruno</PRENOM>
            <url>bmascret@handy.univ-lyon1.fr</url>
        </ETUDIANT>
        …..
    </PROMOTION>
```

L'élément PROMOTION contient des éléments ETUDIANT eux-même composés.

2.2.5 Références d'entités

Une entité peut être vue comme une variable contenant une valeur textuelle.

Les références d'entités des documents XML débutent toujours par une esperluette et se terminent par un point-virgule. Lorsqu'un parseur (on va le définir plus loin) rencontre une

référence d'entité, il va dans la table de symboles créée lors de l'analyse de la DTD (voir plus loin) du document et en extrait la chaîne adéquate, si elle existe. Il substitue cette chaîne à l'instance d'entité.

Les entités servent à représenter de manière claire certains caractères ou chaînes de caractères α est plus facilement compréhensible que ⁈ .

Il est possible de fabriquer ses propres entités.

2.3 Traiter des documents XML

Les document XML sont traités par un composant logiciel nommé parseur, lisant le document XML comme texte brut. Celui-ci implémente en outre une ou plusieurs API (*Application Programming Interface*), comme le modèle objet de document (DOM, pour *Document Object Model*), ou SAX(*Simple API for XML*).

L'API offre aux programmeurs un ensemble de fonctionnalités pouvant être appelées depuis un programme pour réclamer des informations au parseur alors que celui-ci traite le document. Par exemple, un programmeur peut demander au parseur de lui fournir le premier enfant de l'élément racine, ainsi que le texte qu'il contient.

2.4 DTD (Document Type Définition) ou XML Schema ?

2.4.1 Définition de type de document, ou DTD

Dans la spécification XML 1.0, un mécanisme est proposé pour contrôler le contenu pouvant apparaître dans un document XML. Cela s'effectue par une définition de type de document, ou DTD, qui constitue un ensemble de règles devant être respectées par tout document auquel la DTD s'applique.

Certains parseur XML sont capables de valider un document XML par rapport à sa DTD, générant éventuellement une erreur en cas de non-respect des règles.. Si le XML se conforme aux règles de la DTD, il est appelé XML valide, plutôt que simplement bien formé.

Cependant, les DTD souffrent de quelques déficiences. Tout d'abord, les DTD ne sont pas écrites en XML, ce qui signifie que les technologies existantes pour manipuler des documents XML telles que DOM ou SAX ne peuvent être utilisées pour " parser " des schémas de documents.

La syntaxe en elle-même est très limitée. En effet, la syntaxe des DTD repose sur le principe de grammaire limiter en nombre d'occurrences. Il est impossible de définir un nombre précis d'occurrences, ni de définir une plage d'intervalles d'occurrences.

Mais surtout, les DTD n'offrent qu'un typage très limité des données.

Voici par exemple la DTD de notre document XML précédant :

```
<!ELEMENT promotion (etudiant+)>

<!ELEMENT etudiant (nom, prénom, adresse, email+)>

<!ELEMENT nom (#PCDATA)>
```

```
<!ELEMENT prénom (#PCDATA)>
<!ELEMENT email (#PCDATA)>
```

Dans cette DTD, nous définissons l'élément de base *promotion* comme étant composé de un ou plusieurs (+) éléments *etudiant*. L'élément *etudiant* est lui même composé d'un élément *nom*, d'un élément *prénom*, d'un ou plusieurs (+) éléments *email*. Tous ces éléments sont de type texte (PCDATA).

2.4.2 Schémas XML

Conscient des fâcheuses limitations des DTD, le W3C a récemment proposé un nouveau langage de définition de schéma, XML Schema. Conçu pour palier aux déficiences pré-citées des DTD et s'inspirant de la programmation objet, XML Schema propose en plus des fonctionnalités fournies par les DTD :

- un grand nombre de types de données intégrées comme les booléens, les entiers, les intervalles de temps, etc. De plus, il est possible de créé de nouveaux types par ajout de contraintes sur un type existant;
- des types de données utilisateurs qui vous permettent de créer votre propre type de données nommé;
- la notion d'héritage. Les éléments peuvent hériter du contenu et des attributs d'un autre élément. C'est sans aucun doute l'innovation la plus intéressante de XML Schema.
- le support des espaces de nom. Un schéma peut-être vu comme une collection (ou vocabulaire) de définitions de types et de déclarations d'éléments dont le nom appartient à un espace de nom particulier appelé espace de nom cible (target namespace). Il est interdit d'avoir deux éléments de même nom mais ayant deux modèles de contenu différents dans le même espace de nom. Les espaces de nom permettent donc de fournir un contexte à un vocabulaire, ce qui facilite la création de schémas et la validation de documents.
- les indicateurs d'occurrences des éléments peuvent être tout nombre non négatif;
- une grande facilité de conception modulaire de schémas inspirée de l'objet;
- le format utilisé est arborescent, donc parsable.

Pour permettre à certains éléments de contenir des sous-éléments, XML Schema introduit les types de données complexes. Les types de données complexes de XML Schema correspondent aux typedef du langage C ou aux classes des langages objets.

Le mécanisme de contraintes d'occurrences d'éléments de XML Schema est donc beaucoup plus puissant que celui des DTD.

Dans l'exemple qui suit, un élément *etudiant* de type typeEtudiant peut avoir entre 1 et 3 emails :

```
<xsd:complexType name="typeEtudiant">
    <xsd:sequence>
        <xsd:element name="nom" type="xsd:string"/>
        <xsd:element name="prénom" type="xsd:string"/>
```

```
            <xsd:element name="email" type="xsd:string"maxOccurs="3"/>
        </xsd:sequence>
    </xsd:complexType>
```

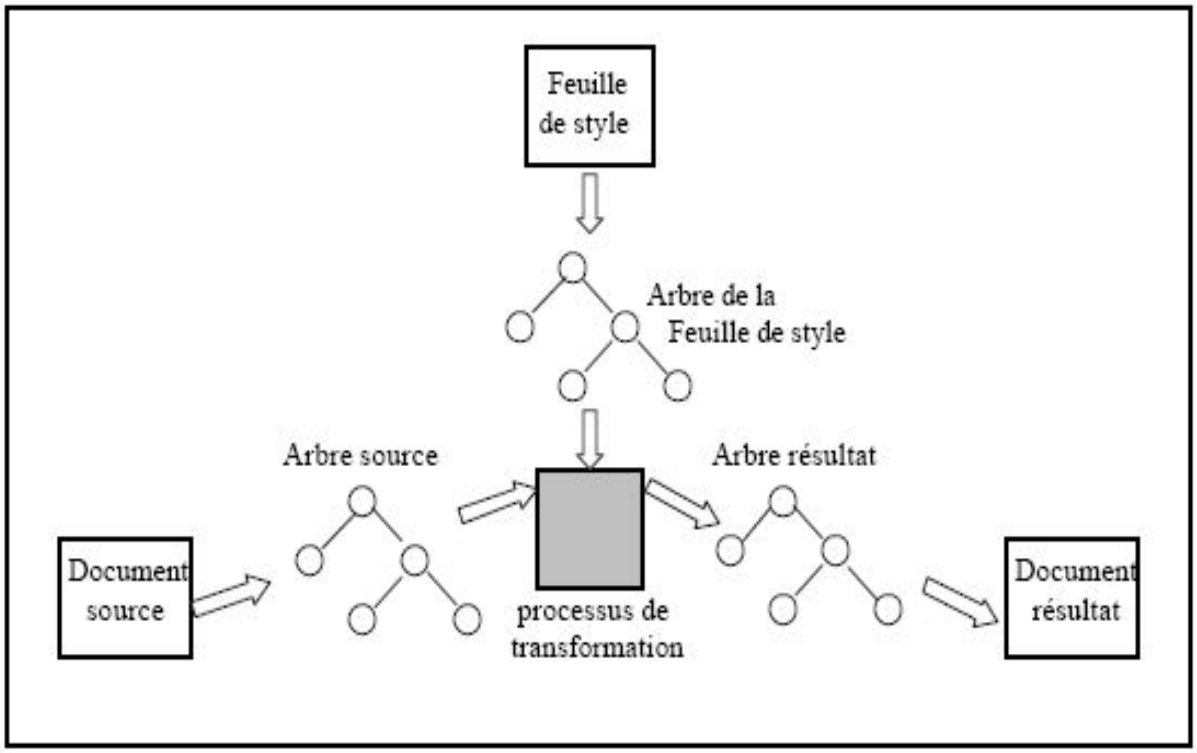
Avec une DTD, le mieux que nous puissions faire est d'indiquer qu'un étudiant possède un ou plusieurs emails (email+).

2.5 Qu’est-ce que XSLT ?

XSLT (e**X**tensible **S**tylesheet **L**anguage **T**ransformations) est un langage de haut niveau qui transforme de la structure d’un document XML en un autre document XML, HTML ou en format basé sur le texte.

2.5.1 Processus de transformation

XSLT a été décrit comme étant un moyen de transformer un document XML en un autre. Cela reste toutefois une simplification. Le diagramme ci-dessous présente ce qui se passe réellement :



Processus d’adaptation

Le processus de transformation comporte trois étapes distinctes :

1. Un parseur XML, part du document source XML et le transforme en une représentation arborescente;
2. Le processus XSLT, suivant les règles exprimées dans une feuille de style, transforme cet arbre en un autre;

3. Un sérialiseur prend l'arbre résultat et le transforme en un document XML..

La transformation est exprimée sous la forme d'un jeu de règles, ces règles reposent sur la définition de la sortie à générer lorsque certains motifs particuliers se présentent à l'entrée.

La plupart des feuilles de style contiennent un certain nombre de règles qui portent le nom de règles modèles. Chaque règle modèle est exprimée dans la feuille de style sous forme d'élément <xsl :template> et possède un attribut *match.* Sa valeur correspond au type de noeud auquel la règle s'applique.

Exemple :

<xsl:template match= "prénom">

Le corps de la règle

</xsl :template>

2.5.2 Processeur XSLT

Lorsqu'on invoque un processeur XSLT pour appliquer une feuille de style particulière à un document, il commence par lire et analyser ce document puis lui crée en mémoire une représentation arborescente interne.

Le rôle principal d'un processeur XSLT est d'appliquer une feuille de style XSLT à un document source et de produire un document résultat, en manipulant les trois arbres présentés plus haut.

Une fois cette préparation achevée, les processus de transformation peuvent débuter.

2.5.3 Variables

XSLT permet de définir des variables globales disponibles dans toute la feuille de style, mais aussi des variables locales, qui ne sont disponibles que dans un corps de modèle particulier. Le nom et la valeur de la variable sont définis dans un élément <xsl:variable>, par exemple :

<xsl:variable name='Var' select='50'>

Ceci définit une variable dont le nom est Var et la valeur égale à 50.Cette variable peut être référencée dans une expression sous la forme $Var. De même XSLT permet de définir les paramètres globaux et locaux à l'aide d'un élément <xsl:param>.

Les types de valeurs qu'on peut affecter aux variables et aux paramètres sont :

- Chaîne ou valeur textuelle;
- Les nombres;
- Booléen (true, false);
- Ensemble de noeuds ou fragments d'arbre.

2.5.4 Expressions

Les expressions sont utilisées dans un certain nombre de contextes dans une feuille de style XSLT. Elles le sont comme valeurs d'attribut pour de nombreux éléments XSLT, par exemple:

<xsl:value-of select='($X+ $Y)*2'>

$X et $Y : références aux variables X et Y.

+ et * : opérations d'addition et de multiplication.

Ces expressions sont connues sous le nom de " expression XPATH " car une expression définit un chemin de navigation sur l'arbre du document.

Outre le fait de spécifier la direction de la navigation dans l'arbre, chaque étape d'une expression XPATH peut également qualifier les noeuds devant être sélectionnés..

La syntaxe d'une expression Path utilise « / » en qualité d'opérateur pour séparer les étapes successives.

Par exemple :

child::item/attribute::category est une expression Path en deux étapes, la première sélectionnant tous les éléments enfant <item> du noeud courant, et la seconde sélectionnant leurs attributs category.

2.5.6 Types de données

XSLT est un langage de type dynamique, dans la mesure où des types sont associés aux valeurs plutôt qu'aux variables. XSLT définit cinq types de données pour effectuer des conversions explicites. Le tableau présenté ci-dessous récapitule les conversions entre les cinq types de données :

	BOOLEEN	NOMBRE	CHAINE	ENSEMBLE DE NOEUDS	OBJET EXTERNE
Booléen	non applicable	faux => 0 vrai => 1	faux => 'faux' vrai => 'vrai'	non admis	non admis
Nombre	0 => faux autre => vrai	non applicable	convertir en format décimal	non admis	non admis
Chaîne	nul => faux autre => vrai	analyser comme nombre décimal	non applicable	non admis	non admis
Ensemble De nœuds	vide => faux autre => vrai	convertir via une chaîne	valeur textuelle de premier nœud dans l'ordre du document	non applicable	non admis
Objet externe	non admis	non admis	non admis	non admis	non applicable

Types de données XSLT

2.5.7 Instructions XSLT

Les instructions définies en XSLT 1.0 seront examinées en détail dans l'annexe B.

2.6 Avantages de XML

XML présente les avantages essentiels suivants :

- XML est libre de droits et indépendant des plate-formes.
- Sa structure hiérarchique (en forme d'arbre) permet de représenter des données structurées, aussi complexes soient elles.
- Lisibilité : les balises utilisent des termes assez intuitifs.
- Indépendance du contenu de sa présentation : une balise indique ce que l'information signifie, non pas comment l'afficher. L'information de formatage pour un fichier XML est écrite dans un langage de style et elle est stockée séparément.
- Réduction de la programmation : la plupart des vérifications d'erreurs sur la validité des documents est faite par l'interpréteur.
- Portabilité : XML ouvre la programmation Java et Internet aux fonctionnalités portables et indépendantes du navigateur. XML libère le contenu Internet du navigateur dans la même mesure que Java libère les comportements d'un programme de la plate-forme. Java est une excellente plate- forme pour l'utilisation de XML, et XML est une bonne représentation des données pour les applications Java.
- Réutilisation : XML va permettre de saisir (ou mettre à jour) en une seule fois un contenu (par ex. une bibliographie) et un contenu pur. Autrement dit sans se soucier de la présentation ou des traitements futurs ; sans avoir à saisir des libellés tels que'"auteur", "année de parution", sans avoir à mettre les titres en italique - exactement, donc, à la manière dont on alimenterait une base de données et d'en générer ensuite automatiquement de multiples présentations (en tableau, en texte suivi…) avec éventuellement tris, sélections, réorganisations, génération automatique de libellés, tables des matières, index, etc. ; et ce sur de multiples médias (écran, papier, terminal braille, etc.). Tout ceci est rendu possible par l'indépendance du balisage par rapport à la présentation.

Le couplage d'un fichier source XML avec un processeur XSLT jouant le rôle de transcodeur semble donc une solution efficace et rapide pour notre transcripteur.

La richesse des instructions XSL et la liberté qu'offre XML devrait permettre l'élaboration d'un processus efficace intégrable dans un programme Java.

Chapitre 3 : Modélisation et Conception

Ce chapitre présente la modélisation des fonctionnalités du système et l'organisation des différents modules entre eux. Les schémas utilisent le formalisme du langage de modélisation UML (cas d'utilisation élaborés et diagramme de classes).

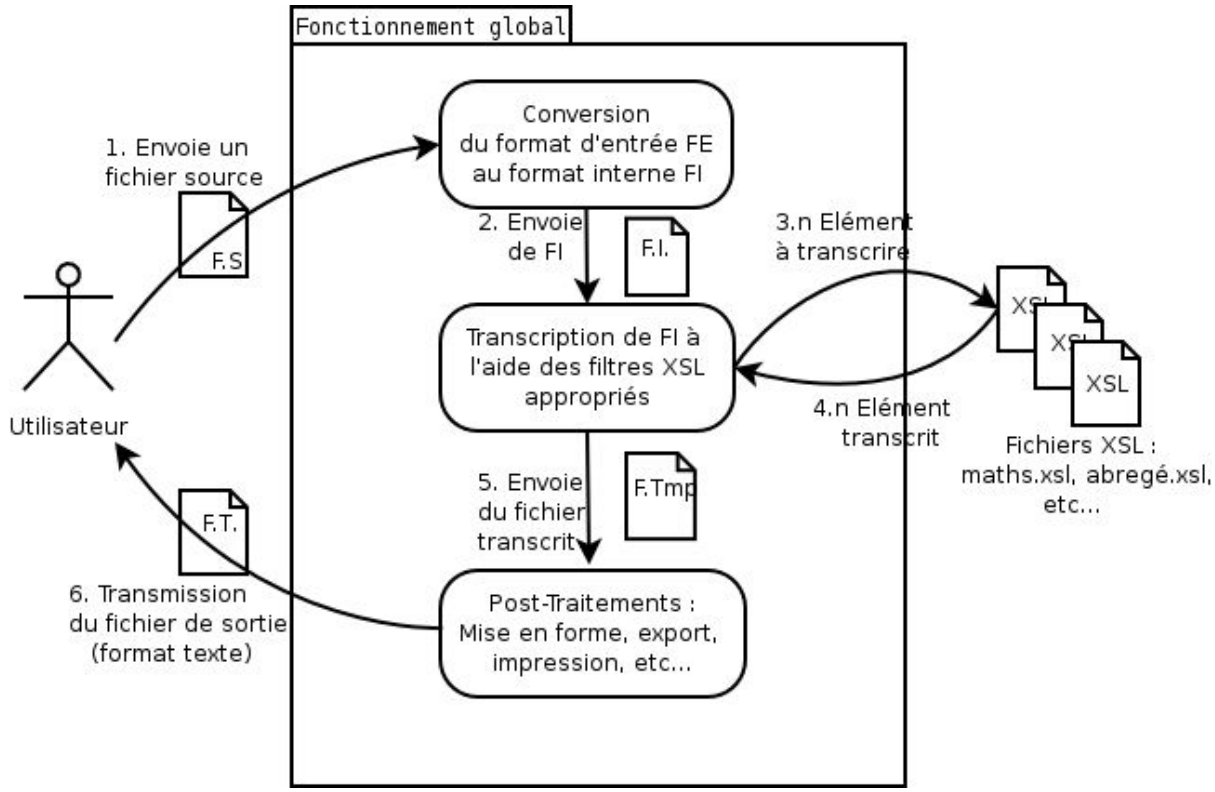
Nous terminons par l'explication des choix des environnements de développement puis d'exécution et des technologies utilisées.

3.1 Fonctionnement général du système

Nous présentons dans cette partie l'organisation des différents modules entre eux ainsi que les principaux formats et flux de données.

3.1.1 Architecture générale du système

Afin de répondre efficacement aux contraintes de simplicité d'utilisation, nous souhaitons minimiser au maximum les interactions de l'utilisateur avec le système. Dans l'idéal, l'utilisateur se contentera de donner un fichier source (F.S.) et obtiendra en retour un fichier transcrit (F.T.).



Nous avons identifié trois modules principaux dans la chaîne de traitement du fichier source :

- **le module de conversion** dont le rôle consistera à transformer le fichier source en un fichier au format interne;

- **le module de transcription** qui effectuera le transcodage du fichier source en braille;
- **le module de post-traitement** qui sera chargé des aménagements à apporter au fichier transcodé afin de le rendre directement exploitable par l'utilisateur selon ses exigences.

3.1.2 Principaux formats et flux de données

Le système sera amené a effectuer de nombreuses transformations tout au long de la chaîne de traitements. Voici les principaux fichiers et formats qu'il utilisera :

- F.S. : Fichier Source ou fichier d'entrée (format texte ou binaire). Ce fichier est transmis directement au logiciel par l'utilisateur et contient des données susceptibles d'être transcrites en braille (typiquement, un fichier provenant d'un logiciel de traitement de texte);
- F.I. : Fichier au format Interne (format texte, en XML). Il permet de présenter les données des formats hétérogènes d'entrée de la même manière, et ainsi de ne pas complexifier la transcription;
- XSL : Fichier XSL filtre (format texte, en XSL);
- F.Tmp : Fichier transcrit, avant post-traitement (format texte). C'est le résultat du transcodage de FI par les filtres XSL.;
- F.T. : Fichier transcrit (format texte). C'est le fichier de sortie remis à l'utilisateur.

Le logiciel doit être capable de prendre en compte un très grand nombre de format de fichiers d'entrée. Il manipulera donc des fichiers sources textes (documents textes, XML, HTML...) ou binaires (documents provenant de logiciels spécifiques d'édition comme Openoffice Writer ou Microsoft Word, ou de formats standards comme RTF).

En revanche, les modules de transcription et de post-traitement n'utiliseront en entrée qu'un seul type de fichiers (formats internes).

En sortie, le module de post-traitement pourra produire différents types de fichiers texte suivant sa configuration. La possibilité de produire des fichiers binaires n'est pas à exclure pour les développements futurs : certaines embosseuses ou imprimantes pourraient ainsi recevoir directement du code machine.

3.1.3 Technologies utilisées lors de la transcription

Le mécanisme de transcription se base sur les outils de transcodage qu'offre le processeur XSLT : à chaque type de contenu, nous ferons correspondre un filtre XSL chargé du transcodage spécifique.

Le choix de cette technologie et son utilisation exacte sont détaillées plus bas.

Afin d'intégrer facilement XSLT dans le processus de transcription, nous avons à notre disposition plusieurs API et plusieurs langages.

Cependant, JAVA propose le plus grand choix d'implémentation robustes d'XML et de XSL (les API SAX et DOM par exemple). Nous intégrerons donc probablement nos feuilles de style grace à une de ces API.

3.2 Fonctionnement détaillé des modules du système

Nous présentons de manière exhaustive dans cette partie les différents modules identifiés précédemment et terminons par la définition des options à prendre en compte.

3.2.1 Conversion du format d'entrée

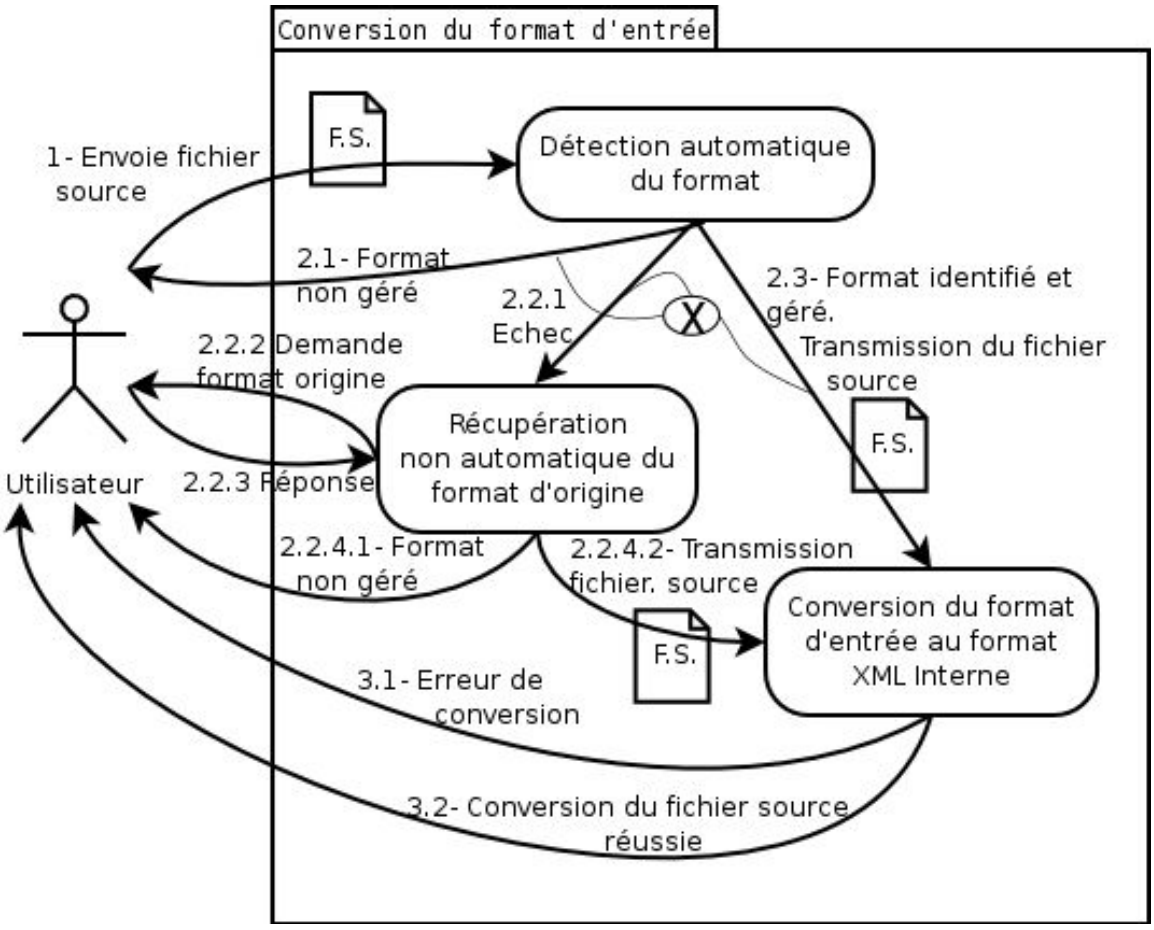
L'objectif de ce module est de convertir le fichier source au format interne.

En effet, il semble plus facile de gérer dès le début des traitements la multiplicités des formats d'entrée possibles dans un module spécifique, plutôt que de déléguer aux modules suivants la charge d'adapter leurs traitements en fonction des différents formats.

L'utilisation d'un format interne offre deux intérêts majeurs :

- garantir au module de transcription un format unique et syntaxiquement correct puisqu'il résulte d'un traitement en amont;
- disposer d'un format créé sur mesure par le système, donc offrant s'il est bien conçu la meilleure présentation possible des données pour les traitements à effectuer.

L'inconvénient d'une telle organisation est qu'il faut prévoir pour chaque type de format d'entrée un traitement spécifique. Devant la multiplicité des formats possibles, il conviendra donc dans un premier temps de ne prendre en compte que les plus utiles et les plus représentatifs d'entre eux.



La conversion du document d'entrée peut générer trois types d'erreurs :

- le format du fichier n'est pas géré : l'erreur est alors bloquante;
- le format du fichier n'est pas reconnu automatiquement : il faudra alors demander à l'utilisateur de préciser quel format il utilise, l'erreur n'est pas bloquante;
- la conversion du document ne s'est pas déroulée correctement (cas d'un fichier endommagé par exemple ou mal détecté) : l'erreur est bloquante.

Une erreur bloquante entraînera nécessairement la fin du traitement et sera notifiée à l'utilisateur.

De même, la réussite de la conversion devra être confirmée, même si 'utilisateur n'a pas à intervenir dans la suite des opérations.

3.2.2 Transcription

Le module de transcription gère à n'en pas douter les fonctionnalités principales de notre logiciel. C'est logiquement le module le plus complexe, tant au niveau des fichiers manipulés qu'au niveau des traitements à réaliser.

Nous reviendrons donc sur les formats et les flux de données spécifiques à ce module après en avoir clarifier le fonctionnement.

3.2.2.1 Fonctionnement général

Le module de transcription comporte quatre phases successives réparties en cinq sous-modules :

- **la phase de configuration** se charge de la récupérations des options de transcription;
- **la phase d'initialisation** prépare la transcription en collectant les filtres utiles et en créant un scénario de transcription;
- **la phase de transcription** réalise le transcodage en appliquant le scénario;
- **la phase terminale** rédige le rapport de transcription et informe l'utilisateur sur le déroulement des traitements effectués.

La phase d'initialisation permet une adaptation fine au fichier à transcrire plutôt que d'appliquer systématiquement le même traitement : supposons qu'un document ne contienne que des écritures littéraires, il serait inutile de charger les filtres de conversion des mathématiques, de la chimie, ou de la musique.

De même le programme s'adapte dynamiquement à la configuration de l'utilisateur.

La mise en place du scénario permet d'une part d'optimiser le logiciel, d'autre part de le rendre facilement paramétrable. C'est le scénario qui déterminera les traitements à effectuer en fonction du contexte de transcription.

De plus, certains filtres sont à appliquer prioritairement suivant les configurations (par exemple, l'abrégé mathématique est prioritaire sur l'abrégé littéraire).

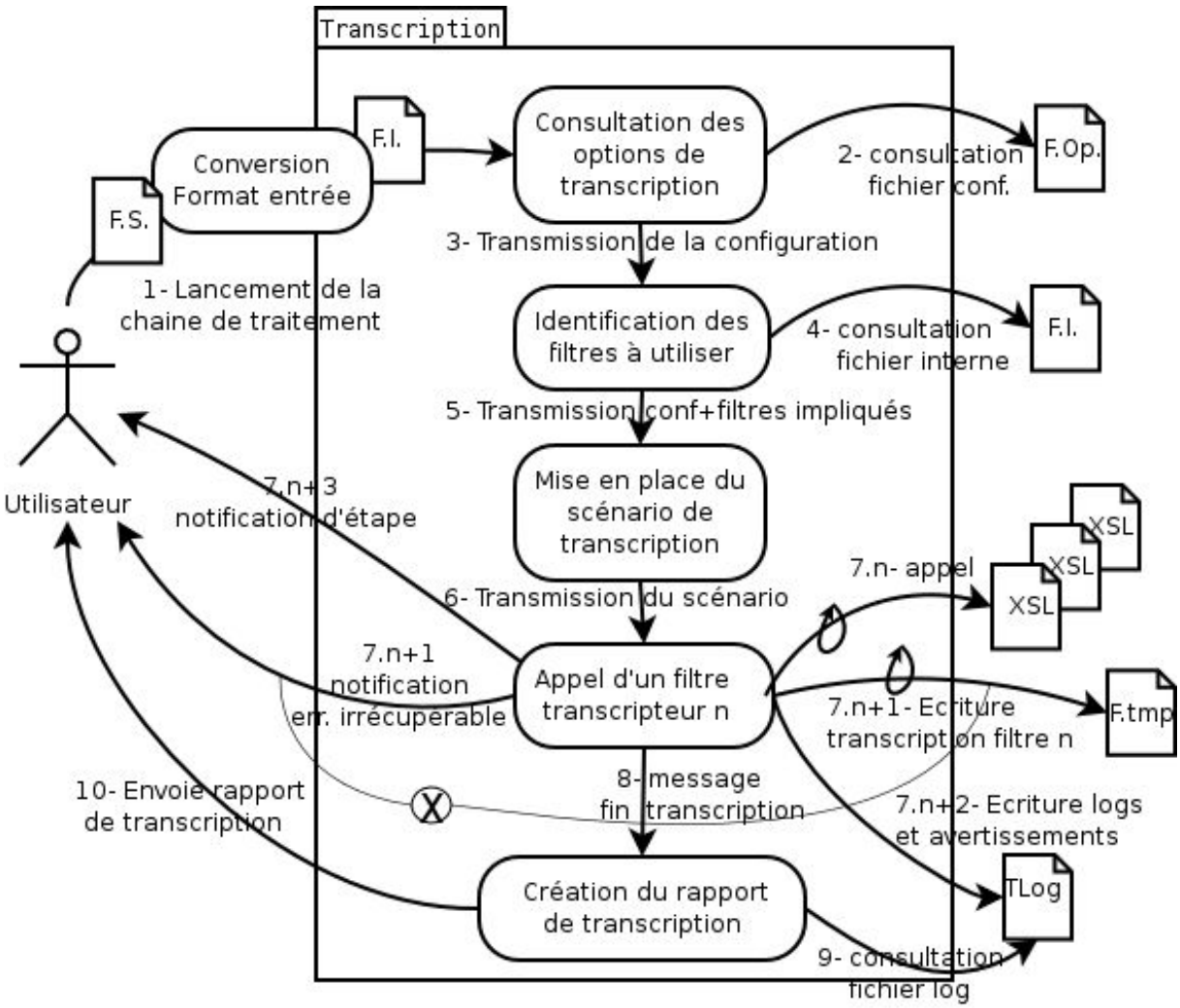
Ainsi préparée, la phase de transcription n'a plus qu'à suivre le scénario à la lettre. Il est d'autant plus important d'alléger cette étape qu'elle reste la plus difficile à implémenter techniquement.

Le rapport de transcription permet à l'utilisateur d'obtenir un retour sur le déroulement de cette étape. Ce rapport sera à adapter en fonction des exigences de l'utilisateur : un transcripteur professionnel souhaitera un rapport fonctionnel détaillé, un développeur préférera un rapport technique exhaustif alors qu'un utilisateur occasionnel ou peu compétent n'aura besoin que des grandes lignes.

Deux types d'erreurs peuvent être générés par le processus de transcription :

- des erreurs bloquantes (par exemple un code de caractère non reconnu ou une entité non prise en compte dans un filtre) : la transcription est alors arrêtées;
- des erreurs non-bloquantes (ambiguïté dans une transcription, plusieurs choix de transcriptions possibles, etc.) : ces erreurs seront notifiées si l'utilisateur l'a demandé et consignées dans le rapport.

L'utilisateur doit s'il le désire recevoir les informations sur le déroulement de la transcription en direct, on lui notifiera donc les différentes étapes du processus. Toutefois, il ne sera pas en mesure d'interagir avec le système durant la procédure si tout se déroule normalement.



3.2.2.2 Description détaillée des fichiers impliqués

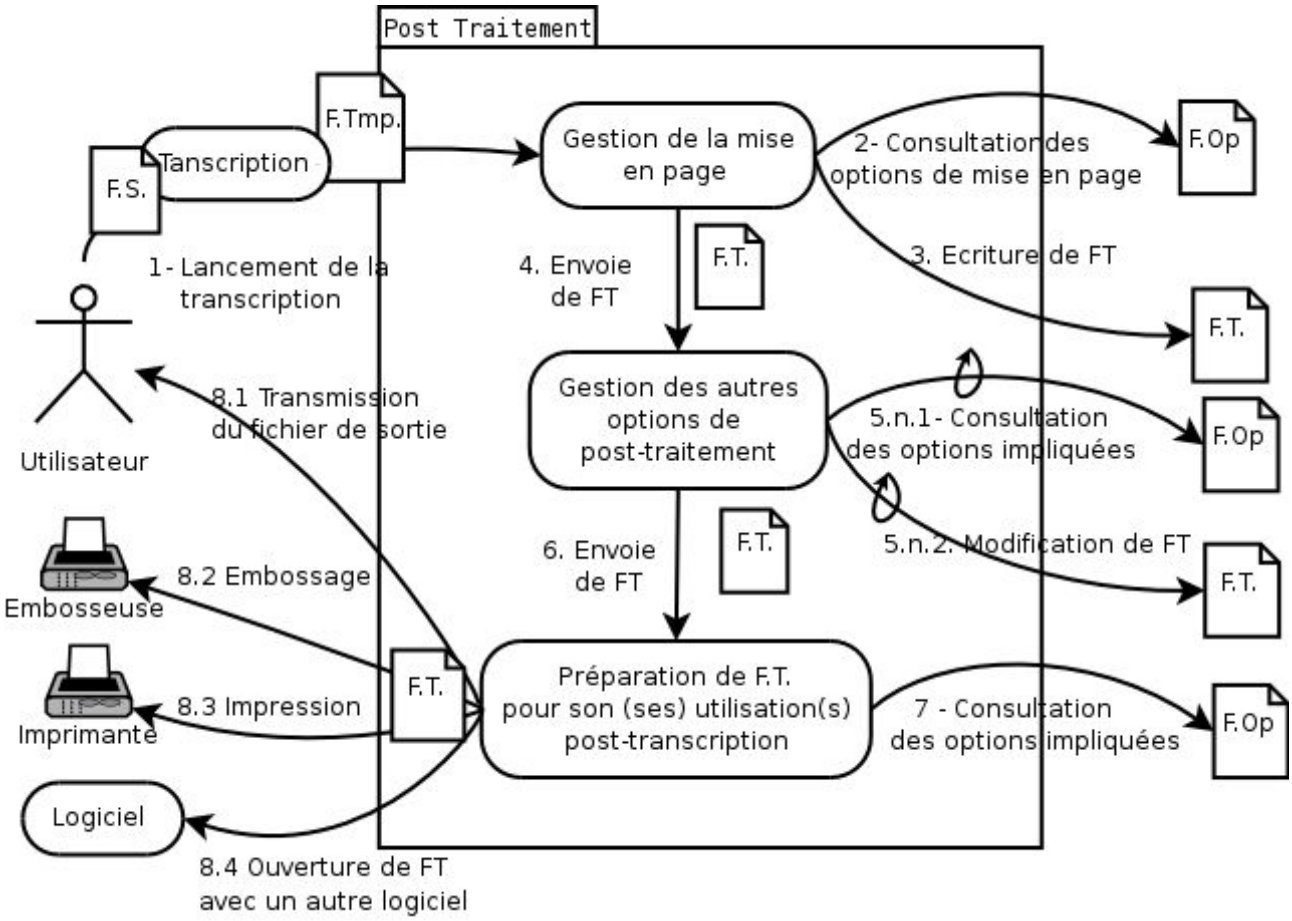
- F.S. (Fichier Source), F.I. (Fichier au format Interne), XSL (Fichier XSL filtre), F.Tmp (Fichier transcrit, avant post-traitement) : voir plus haut;
- F.Op : Fichier d'Options, au format texte (possibilité d'utiliser un document XML). Il stocke les préférences des utilisateurs et la configuration en cours.
- Tlog : Fichier de logs de la transcription (format texte). Il contient l'ensemble des messages générés lors de la transcription par le processeur XSLT et par le logiciel. Il doit distinguer les différents types de messages afin de pouvoir identifier l'origine, le niveau de détail et l'importance de chaque message.

3.2.3 Post-Traitements

Le module de post-traitement à plusieurs objectifs :

- s'assurer de la bonne présentation du document, en représentant par exemple les structures (titre, listes,...) et les mises en formes (alignements, indentations,...).
- préparer le document afin de le rendre compatible avec le mode de sortie spécifié par l'utilisateur (embossage, impression, lecture sur plage braille, traitement avec un autre logiciel comme par exemple un logiciel d'impression spécifique)
- pouvoir intégrer d'autres fonctionnalités de post-traitement non prévues pour l'instant.

Une fois encore, afin de faciliter l'utilisation du logiciel, cette étape sera entièrement paramétrée à l'aide du fichier d'options.



3.2.4 Paramétrages

Un des critères les plus recherchés par les transcripteurs est la facilité d'adaptation du logiciel à leurs besoins spécifiques. Nous devrons donc dans la mesure du possible offrir une grande variété de paramètres tout en restant pertinent.

Les principales options à gérer seront :

- l'édition de la table Braille;
- le choix des filtres à utiliser pour un contenu donné (abrégé ou intégral par exemple);
- la possibilité de choisir les contenus à traiter;
- permettre la configuration indépendante pour chaque filtre;
- maintenir un dictionnaire d'éléments à ignorer lors de la conversion;
- autoriser ou non les coupures (nombre caractères/lignes, coupure mathématique);
- offrir différents niveaux de détail pour le compte-rendu de transcription.

A ces paramètres s'ajoutent encore d'autres réglages très spécifiques ou techniques que nous ne présenterons pas à ce niveau de la modélisation.

3.3 Modélisation de la structure du format d'échange interne

Cette partie détaille la méthode suivie pour réaliser la modélisation du format de document interne. Après une modélisation générique d'un document puis la mise en évidences des différences entre un document noir et un document braille, nous examinons la possibilité d'utiliser des formats existants.

Nous terminons par la spécification de notre format interne.

3.3.1 Modélisation de documents noir et Braille

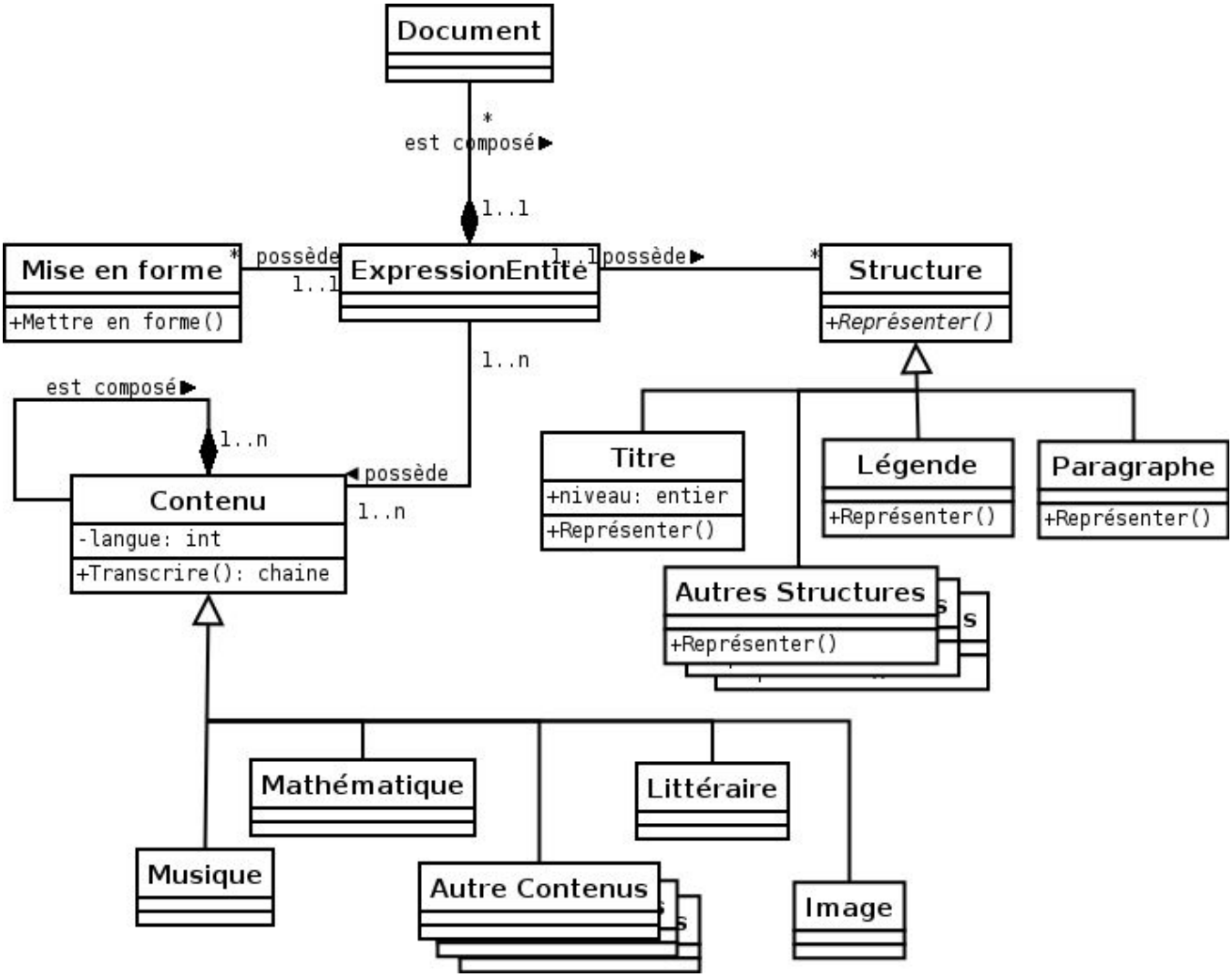
Le diagramme suivant propose une modélisation générique d'un document " en noir "quelconque.

Bien que de nombreuses modélisations de documents existent déjà, nous nous sommes inspirés des différentes réflexions menées sur l'accessibilité informatique aux personnes handicapées pour établir notre modèle.

L'approche choisie lors de la modélisation s'appuie sur la séparation du contenu, de la forme et de la structure d'une expression. Si de nombreuses études prônent déjà la dissociation du contenu et de la forme, nous présentons un niveau supplémentaire d'organisation en distinguant – dans ce qu'on appelle " forme " – structure et présentation d'une expression.

La structure d'une expression lui donne son rôle dans le document : corps de texte, titre, note de bas de page, etc. La présentation d'une expression lui donne son aspect : police de caractère, alignement, etc. Ces deux aspects sont différents par nature : choisir de les traiter indépendamment dans un processus de transcription semble donc logique.

Ce choix nous offre également la possibilité de traiter et de paramétrer séparément chaque caractéristique d'une expression. L'utilisation de filtres XSL spécialisés permettra de réaliser ces opérations.



Modélisation d'un document " en noir "

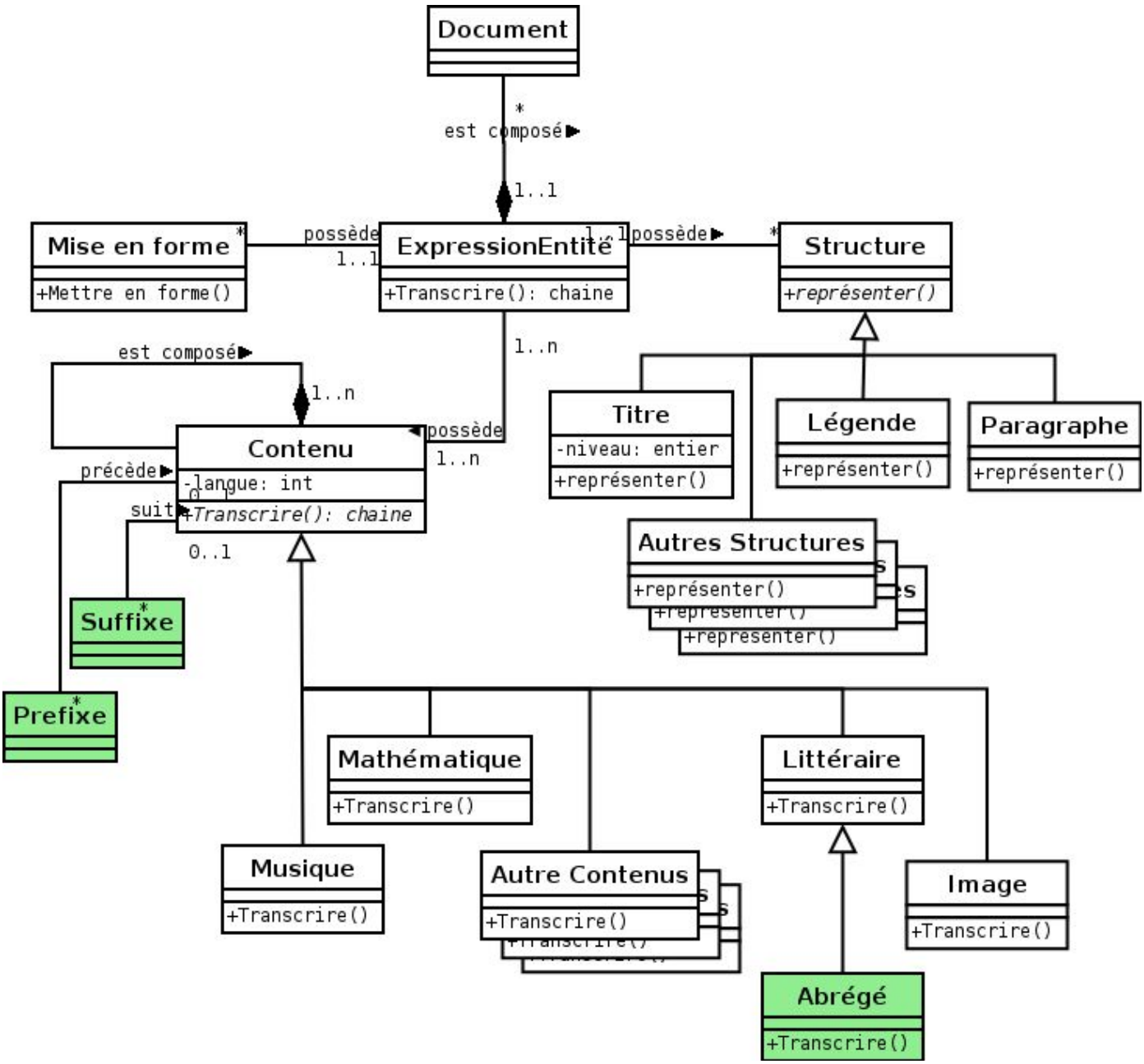
L'autre diagramme (" Modélisation d'un document Braille ") met en évidence les différences qui existent entre un document " en noir " et un document braille.

Contrairement à ce qu'on pourrait penser, le modèle braille obtenu est plus complexe que le modèle noir. Il ne faut pas perdre de vue :

- que notre point de vue de modélisateurs n'est pas objectif, car nous cherchons à mettre en place un transcripteur pour le braille;
- que nous ne modélisons que ce dont nous avons besoin;
- que le braille dérive de formes d'écritures plus anciennes que lui et peut donc être vue comme une amélioration de ces écritures (du point de vue du modèle);
- que le braille est essentiellement linéaire, contrairement à certaines écritures qu'il transcrit qui sont elles bidimensionnelles (mathématique, musique...) : il doit donc intégrer ces projections de la dimension 2 vers la dimension 1.

Ces deux modélisations vont nous servir de point de départ pour le choix d'un format interne compatible avec les deux modèles. La bonne spécification de ce format est cruciale car elle va conditionner les performances en termes de rapidité, de facilité d'implémentation et d'efficacité du logiciel.

Dans un premier temps, notre objectif concerne uniquement le contenu mais nous ne devons pas oublier pour autant les autres composantes d'une entité-expression qui seront pris en compte lors d'une version suivante du logiciel.



Modélisation d'un document en Braille

3.3.2 Réflexions sur les formats existants

Notre recherche de formats déjà existant s'est logiquement concentrée sur les formats XML définis par des DTD ou des schémas XML. Les traitements principaux se basant sur l'application de feuilles de style, autant utilise en entrée un document optimisé pour cette technologie.

Nous avons retenu 3 formats XML particulièrement intéressants :

- le format DTBook;
- le format Daisy;
- le format OpenOffice 1.

Malgré leur nombreuses fonctionnalités, ces formats ne répondent pas suffisement bien à nos attentes. Leur principal défaut est probablement l'utilisation de la phrase comme unité minimale.

Il nous semble préférable de spécifier nous-mêmes un format sur mesure en s'inspirant de nos modèles précédents plutôt que de chercher à rendre compatible un format complexe avec notre logiciel.

Néanmoins, l'utilisation de ces formats comme sorties possibles est à retenir.

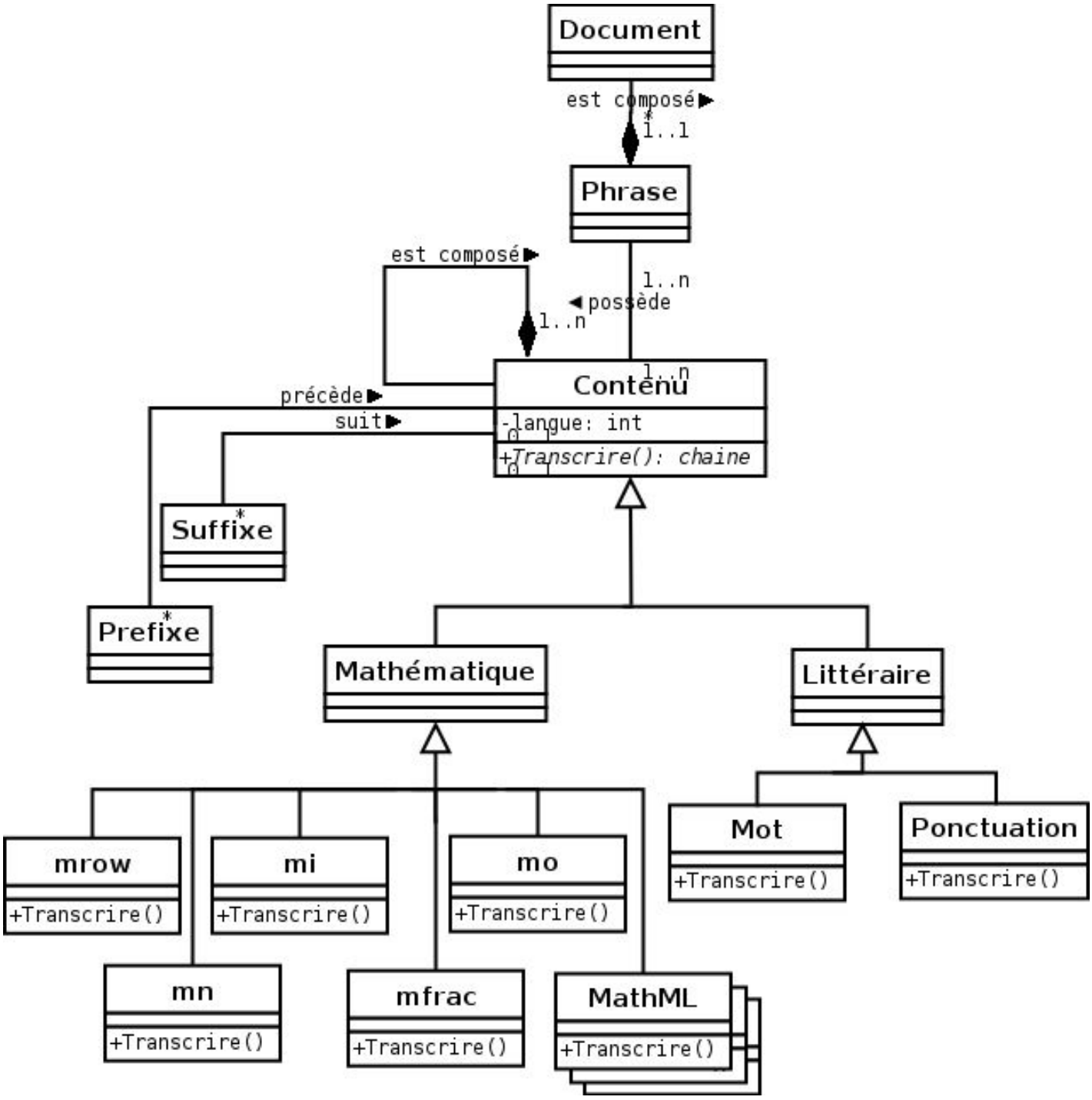
3.3.3 Proposition d'un format interne et justification des choix effectués

Le format interne retenu pour nos besoins actuels est modélisé sur le diagramme de classe suivant.

Nous nous sommes concentrés sur la spécification des contenus, littéraires et mathématiques :

- Nous utiliserons le standard MathML pour les expressions mathématiques;
- Nous utiliserons un format simple pour les expressions littéraires, de niveau suffisement fin pour permettre un traitement mot à mots. Nous distinguerons les signes de ponctuation des mots afin de faciliter l'algorithmie.

Les aspects mise en forme et structure pourront être modéliser ultérieurement, sans remettre en question l'organisation actuelle. Nous avons préféré nous concentrer dans un premier temps sur le contenu.

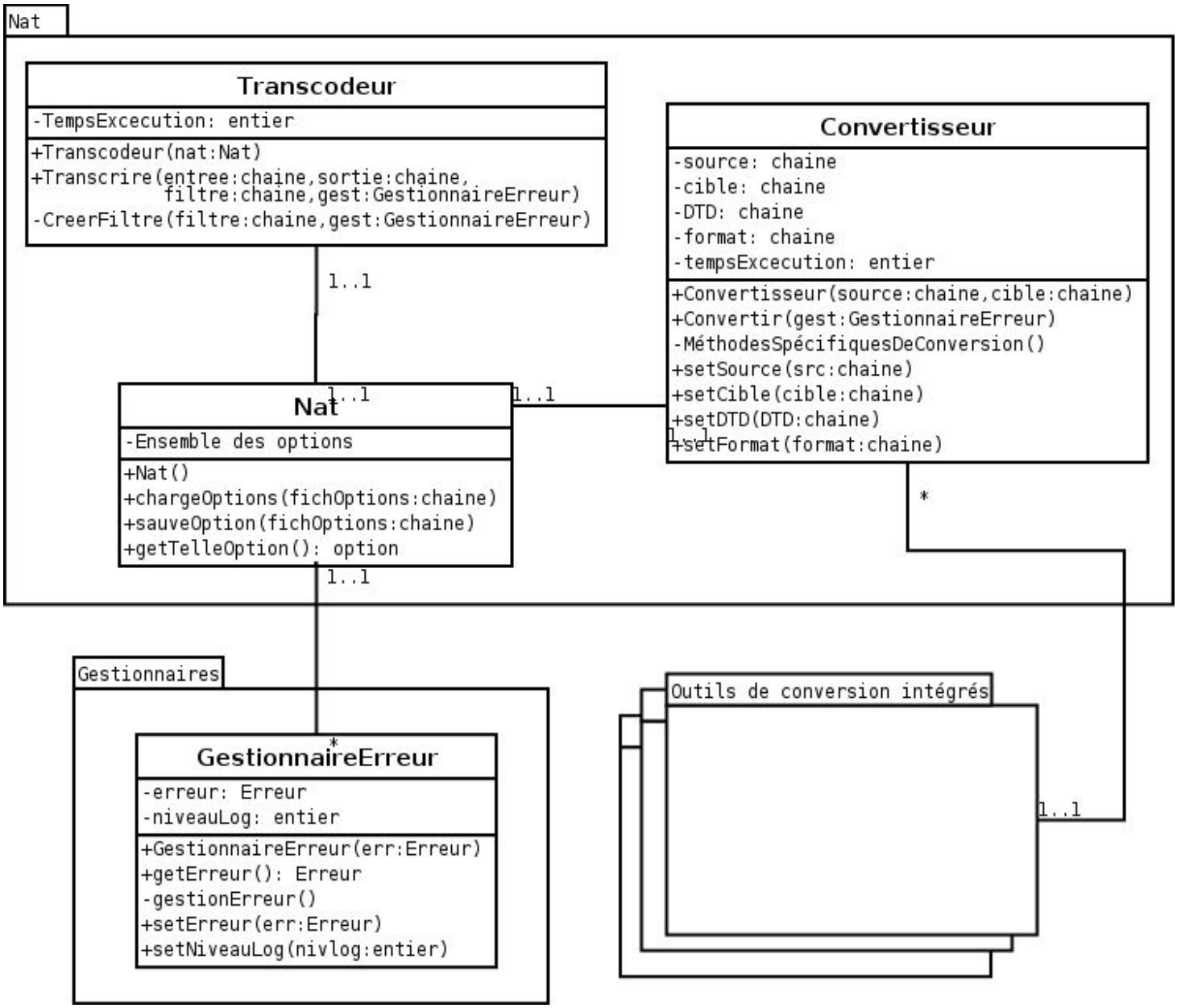


Modélisation du format de document interne.

3.4 Modélisation générale des fonctionnalités du système

Ce chapitre présente la modélisation des fonctionnalités du système qui seront par la suite programmés à l'aide d'un langage objet.

Cette modélisation découle directement de l'analyse des cas d'utilisation précédemment proposés. Elle a pour but d'organiser le développement et l'implémentation des fonctionnalités.



Modélisation des fonctionnalités du système

Le modèle est divisé en trois paquetages distincts :

- le paquetage Nat, qui contient la classe principale (Nat) et les deux classes de traitements (Convertisseur et Transcodeur);
- le paquetage Gestionnaires qui sera sollicité pour la gestion des différents types d'erreurs possibles;
- la collections de paquetages intégrés : ce sont les outils existants que nous utiliseront lors des conversions de documents et qui seront directement liés au convertisseur.

La classe principale Nat sera chargée du paramétrage, les options (attribut Ensemble des options) et leur méthode d'accès en lecture (getTelleOption()) n'ont cependant pas été représentées pour alléger la lecture.

Nous avons choisi une approche fonctionnelle pour notre modèle.

3.5 Modélisation de l'interface graphique du système

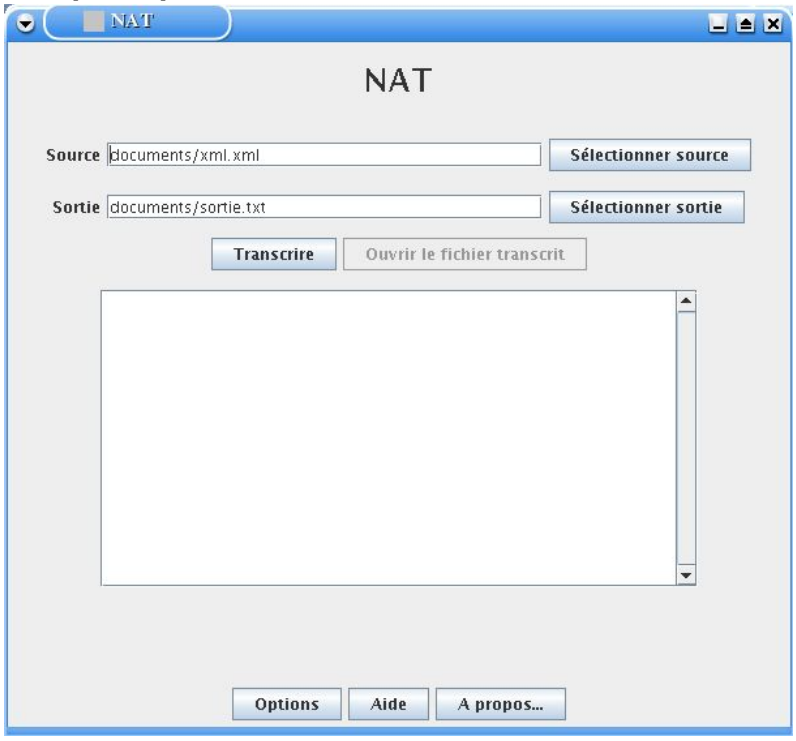
Cette partie propose une ébauche d'interface pour le système en conservant la notation UML pour les mêmes raisons que précédemment.

Après avoir présenté une maquette générale, nous l'organisons dans un diagramme de classe. Nous supposons que les composants graphiques standards (fenêtres, boutons, etc.) seront présents dans l'API du langage de programmation retenu.

3.5.1 Maquette de l'interface

Cette maquette a pour but de nous permettre d'identifier les composants à utiliser et de les organiser entre eux.

3.5.1.1 Fenêtre principale



La fenêtre principale doit permettre à l'utilisateur d'accéder à l'ensemble des fonctionnalités du logiciel, et interfacera le fonctionnement principal du logiciel.

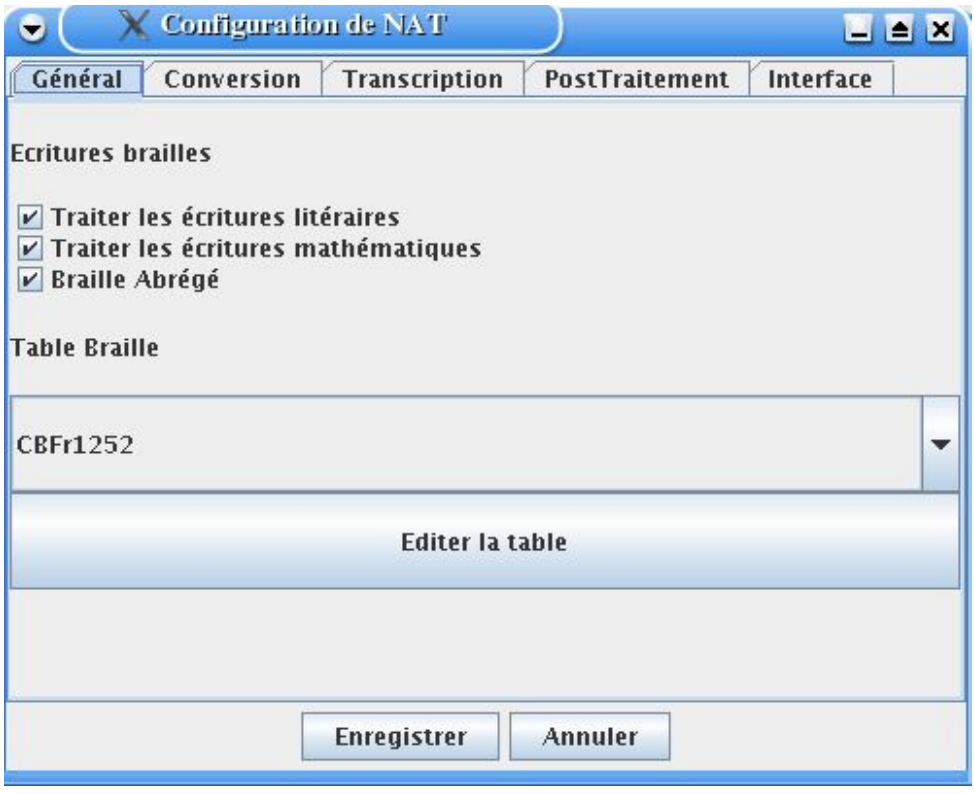
Elle proposera donc :

- une première zone destinée aux entrées (fichiers source/cible) et au lancement des traitements;
- une seconde zone destinée aux sorties (erreurs, état de la transcription, communication);
- une dernière zone permettant l'accès aux autres éléments de l'interface.

3.5.1.2 Fenêtre options

La fenêtre d'option regroupera l'ensemble des paramètres utilisés pour la transcription et la configuration du logiciel.

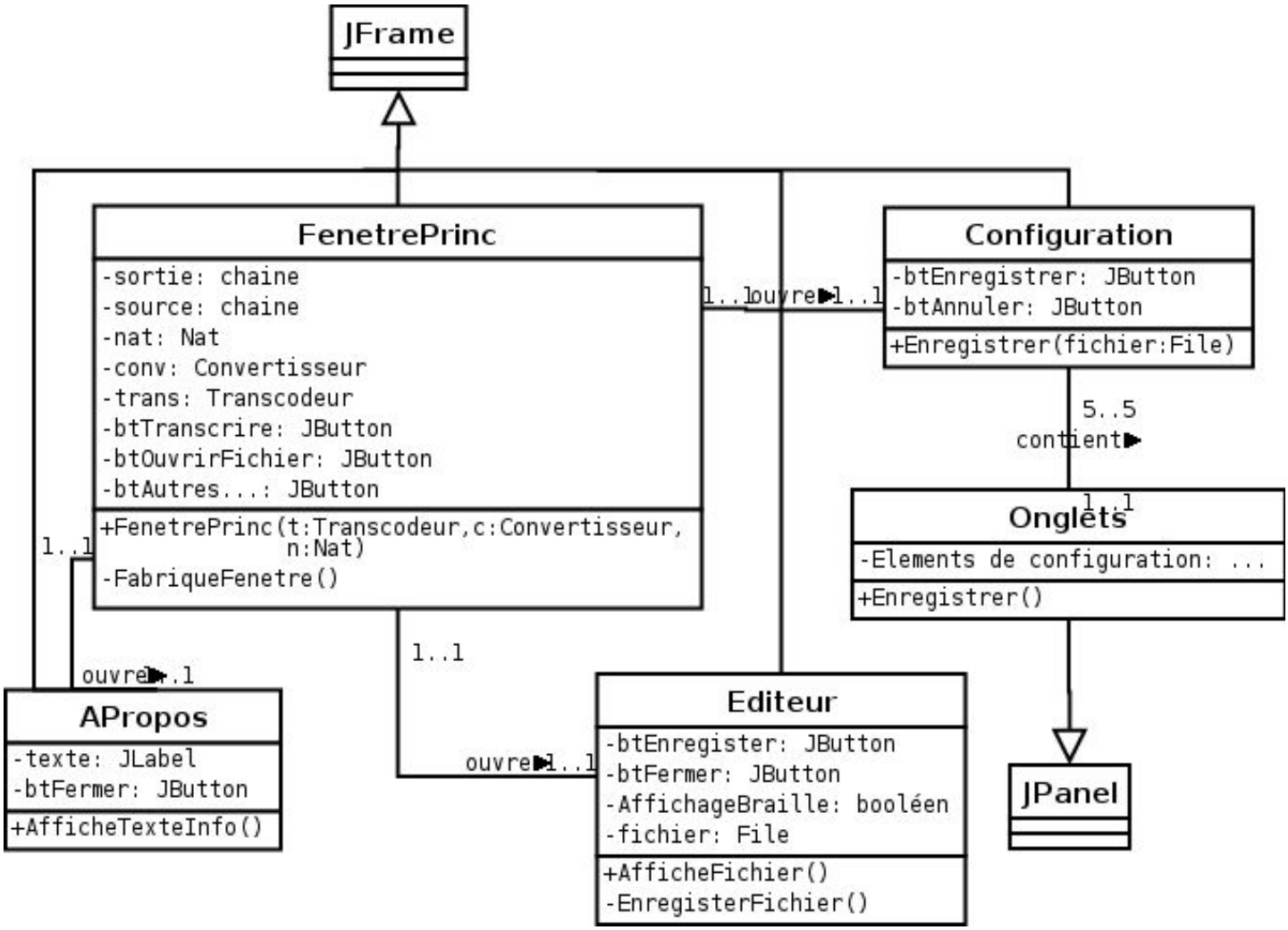
Nous les regrouperons thématiquement par onglets.



3.5.2 Modélisation de l'interface

Le diagramme de classe suivant présente l'organisation générale de l'interface graphique. Les composants standards utilisés sont ceux de l'API Swing de Java, mais pourraient tout aussi bien provenir d'API d'autres langages de programmation.

Les composants des onglets n'ont volontairement pas été détaillés afin de simplifier le modèles et laisser une plus grande liberté d'évolution aux développeurs..



3.6 Modélisation générale

Cette partie présente le modèle général et en justifie l'architecture.

3.6.1 Organisation du système en MVC

Grâce au développement des techniques de programmation et de modélisation, l'axiome " séparer le contenu de la manière de le présenter " s'impose de plus en plus en tant que " Best practice " que l'on pourrait traduire en français par " recommandation pour obtenir un programme bien réalisé ".

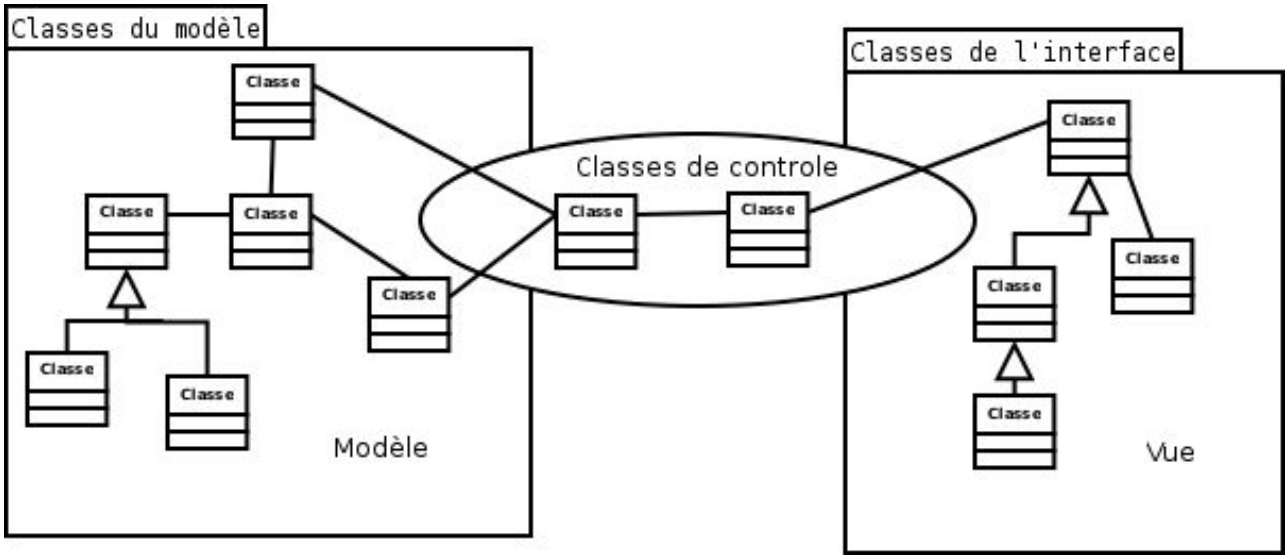
De ce concept est né l'idée de l'architecture MVC (Modèle, Vue, Contrôleur). Cette architecture permet de séparer les fonctionnalités du système de leur interfaçage, la communication entre les deux ensembles étant assurée par les contrôleurs.

Dans notre cas, nous devrons probablement adapter notre interface à différents type de public : il semble d'autant plus judicieux d'utiliser une telle architecture qui nous permettra pour un même ensemble de fonctionnalités de proposer plusieurs interfaces.

L'autre intérêt majeur de ce type d'organisation est la réutilisabilité des différents composants de manière indépendante : en cas d'intégration de notre logiciel, seuls les classes

utiles seront récupérées. De plus, en cas de modifications fonctionnelle ou graphique, seul le package concerné sera modifié sans avoir à toucher aux autres.

Nous incitons le lecteur à se remémorer les critères d'accessibilité du W3C : il s'agit bien plus que d'une simple analogie...



L'architecture MVC

3.6.2 Modèle général

Le modèle général est disponible en annexe (Annexe A).

Il intègre les modèles précédemment proposés ainsi que les classes de contrôle nécessaires au bon fonctionnement du système.

3.7 Implémentation

Cette partie présente les choix techniques retenus pour l'instanciation de notre modèle et de ses environnements de développement et d'exécution.

3.7.1 Choix des environnements

3.7.1.1 Environnements de développement

Les développements seront effectués sous Linux (Debian Sarge [WEB 16]) à l'aide d'Eclipse [WEB 14] pour la programmation Java et de l'éditeur texte Kate [WEB 15].

Une partie du développement des feuilles de style a néanmoins été réalisé sous Microsoft XP à l'aide de l'éditeur texte NotePad.

3.7.1.2 Environnements d'exécution

L'objectif de notre logiciel est de servir au plus grand nombre sans trop contraindre les utilisateurs à adopter une configuration logicielle spécifique.

L'utilisation de technologies ouvertes et libres ainsi que de Java devrait permettre à n'importe quel système d'exploitation doté d'une machine virtuelle Java récente de faire fonctionner notre logiciel.

3.7.2 Choix des outils de programmation

L'étude préalable nous incite fortement à utiliser une solution XML-XSL interfacée par le langage de programmation JAVA.

Nous avons donc choisi les outils suivants :

- programme principal : Java
- transcription : appel à des feuilles XSL grace aux API DOM et SAX de Java
- interface graphique : API Swing de Java

Chapitre 4 : Développements

Ce chapitre présente les développements réalisés et les travaux associés.

4.1 Le module de transcription

4.1.1 Définition de la DTD

Bien que le XML Schema offre plus de fonctionnalités que la DTD, nous avons préféré dans un premier temps nous contenter d'une structuration par DTD, plus simple à mettre en place et surtout mieux interprétée par les différentes API.

Nous nous sommes focalisés sur l'aspect contenu, reléguant la définition des deux autres aspects d'une expression aux prochaines versions du logiciel.

```
<!-- DTD pour NAT -->
<!-- pour les conversions venant de xhtml -->
<!ENTITY % laDtdXHTML SYSTEM "xhtml11.dtd" >
%laDtdXHTML;
<!-- pour le contenu mathÃ©matique -->
<!ENTITY % laDtdMath SYSTEM "mathml.dtd" >
%laDtdMath;
<!-- pour le contenu litÃ©raire -->
<!ELEMENT mot (#PCDATA)>
<!ELEMENT ponctuation (#PCDATA)>
<!ELEMENT lit ((mot | ponctuation)*)>
<!-- Element conteneurs de contenus -->
<!ELEMENT phrase ((math | lit)*)>
<!-- Element de base -->
<!ELEMENT doc (phrase*)>
```

Bien que courte en apparence, cette DTD inclut les DTD XHTML1.1 et MATHM, soit plus de 3000 lignes de définitions...

4.1.2 Intégration de la feuille de style mathématique de BraMaNet

Pour la gestion des mathématiques, nous avons repris et amélioré la feuille de style BraMaTxt.xsl du logiciel libre BraMaNet.

Cette feuille de style, parfaite pour le MathMl généré pas MathType, présentait un grand nombre de lacunes pour s'adapter à la norme officielle MathMl.

Plutôt que de nous enfermer strictement dans la norme, nous avons préféré adapter cette feuille de style pour qu'elle reste compatible avec les différents MathMl existants.

Nous l'avons également augmentée d'expressions et de symboles non-traités jusqu'alors.

4.1.3 Création des feuilles de style intégral et abrégé pour le littéraire

Nous avons commencé par implémenter la feuille de style pour le braille intégral. Sa programmation quasi-trivial n'a pas posé beaucoup de problèmes techniques et nous a permis de régler rapidement les soucis de compatibilité de la feuille mathématique avec un autre filtre.

En revanche, la conception de la feuille de style pour l'abrégé s'est révélée beaucoup plus complexe. En effet, nous n'avons pas souhaité traiter l'abrégé par un immense dictionnaire, ce qui aurait été plus facile mais beaucoup moins évolutif et performant.

Après l'analyse des règles d'abréviation, nous les avons transcrites en XSL et ordonnées logiquement grace à l'algorithme de la page suivante. Les zones grisées correspondent à un contenu à analyser ou a une modification de ce contenu et les points cerclés à un état final (la transcription a été trouvée).

L'abrégé comprend deux ensembles de règles : les règles d'exception et les règles du cas général. Le temps imparti par le calendrier ne nous permettant pas de traiter l'ensemble des cas possibles, nous avons commencé par les règles d'exception, puisque ce sont elles qui s'appliquent en premier.

Les numéros utilisés dans l'algorithme correspondent à la numérotation des règles utilisée dans le document officiel de l'A.V.H. " Le braille abrégé " que l'on peut se procurer sur le site internet de l'association (http://www.avh.asso.fr) ou en écrivant au service commercial.

Une fois ces règles implémentées, nous avons optimisé les traitements en étudiant les fréquences d'apparition des mots-exceptions dans la langue française. Le classement alphabétique a donc été remplacé par un classement fréquentielle.

Malgré nos efforts, certaines ambiguïtés subsistent, qui ne pourront être levées que par une analyse étymologique du mot. Une partie de cette analyse est réalisable simplement par informatique, mais l'analyse exhaustive est elle très complexe à implémenter.

Les prochaines versions du logiciel devront être capables de détecter une ambiguïté et demanderont dans ce cas à l'utilisateur de choisir la bonne transcription parmi plusieurs propositions.

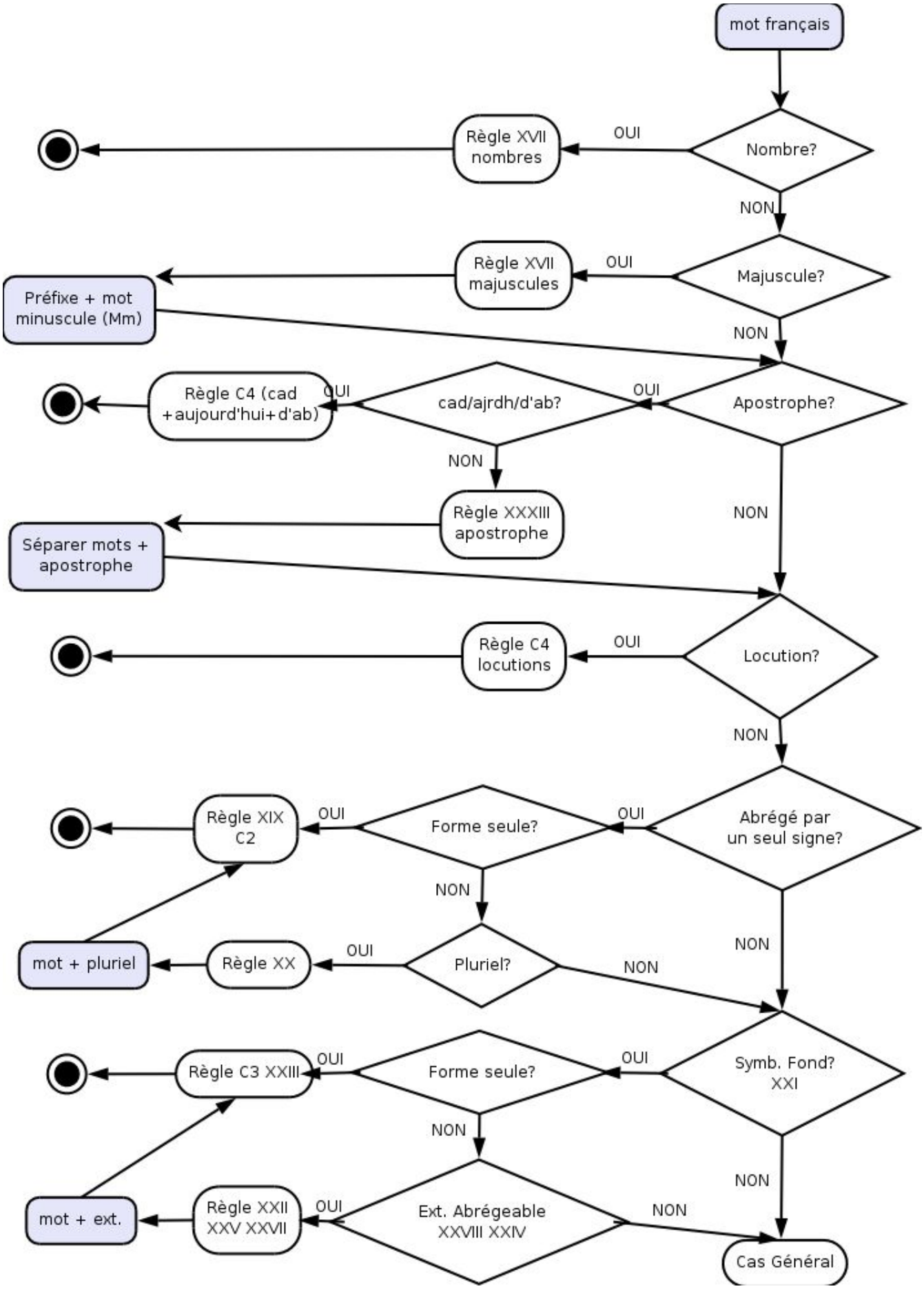
La même méthode de travail devra être appliquée pour la réalisation du cas général.

4.1.4 Implémentation du module avec Java

L'utilisation des API SAX et DOM, bien que théoriquement triviales, s'est révélées un peu plus ardue que prévue. En effet, dans notre soucis de compatibilité inter-plateforme, nous avons dû implémenter exhaustivement les interfaces, travail fastidieux et complexe.

Le résultat obtenu est très satisfaisant, comme l'on montré les premiers tests unitaires.

Afin de maîtriser l'ensemble du processus, nous avons identifié chaque erreur possible (Exception de Java) et la traitons de manière appropriée. L'utilisateur disposera donc d'un outil performant et compréhensible de gestion d'erreurs, et pourra suivre le fil des traitements très simplement.


mot français
Nombre?
OUI
Règle XVII nombres
NON
Majuscule?
OUI
Règle XVII majuscules
Préfixe + mot minuscule (Mm)
NON
Apostrophe?
OUI
cad/ajrdh/d'ab?
OUI
Règle C4 (cad +aujourd'hui+d'ab)
NON
Règle XXXIII apostrophe
Séparer mots + apostrophe
NON
Locution?
OUI
Règle C4 locutions
NON
Abrégé par un seul signe?
OUI
Forme seule?
OUI
Règle XIX C2
NON
Pluriel?
OUI
Règle XX
mot + pluriel
NON
Symb. Fond? XXI
OUI
Forme seule?
OUI
Règle C3 XXIII
NON
Ext. Abrégeable XXVIII XXIV
OUI
Règle XXII XXV XXVII
mot + ext.
NON
Cas Général

4.2.1 Conversion d'un fichier texte

Le format le plus simple qui puisse exister est le format texte. Il est fréquemment utilisé par les non-voyants (mels , sortie de scanner à reconnaissance de forme, etc.) et c'est donc logiquement celui-ci que nous avons traité en premier.

La conversion au format interne a été réalisée uniquement en Java, le format d'entrée restant simple à appréhender.

Afin de conserver la compatibilité avec les fichiers exportés par MathType, nous avons également intégré une fonction de reconnaissance du MathMl. Un fichier préparé pour BraMaNet reste donc utilisable avec NAT.

4.2.2 Conversion d'un fichier open-office

La conversion open-office est bien plus difficile à réaliser. Nous avons choisi d'intégrer un logiciel libre, Writer2XHTML, permettant d'exporter proprement un fichier open-office en XHTML.

Malheureusement, la définition des entités mathématiques laissant fortement à désirer, nous avons donc reprogrammé l'ensemble du module chargé de gérer les formules open-office.

Une fois ce travail réalisé, nous obtenions un fichier XHTML correct incluant correctement des expressions en MathMl. Afin de le transformer en document interne, nous avons utilisé de nouveau un filtre XSL dédié.

4.3 Les interfaces homme-machine

4.3.1 Le mode console

Nous avons voulu permettre à un utilisateur d'utiliser NAT uniquement en mode console. Les intérêts de cette idée sont multiples :

- un utilisateur peut préférer le mode console pour diverses raisons (système d'exploitation en mode texte, connexion à un serveur par protocole texte comme ssh ou telnet, etc.);
- le logiciel pourra être exécuté par un autre programme, ce qui facilite son intégration dans d'autres outils;
- le logiciel devient compatible Linux, et pourra être incorporé dans les distributions de ce système d'exploitation libre.

Cependant, le mode console ne permet pas un paramétrage facile, puisqu'il faudra modifier directement les fichiers de configuration sans passer par une interface. Une évolution possible du logiciel serait la mise en place d'un outil de configuration textuel.

4.3.2 Le mode graphique

Le mode graphique offre plus de fonctionnalités que le mode console, notamment en ce qui concerne le paramétrage du logiciel.

Il a été réalisé à l'aide du paquetage SWING de Java, contenant des composants légers et accessibles pour les lecteurs d'écrans. Nous avons également implémenté l'interface d'accessibilité proposée par Java.

Les résultats de l'étude ergonomique étant arrivés trop tard (voir 4.5), un certain nombre d'éléments laissent encore à désirer, même si l'interface permet l'utilisation de toutes les fonctionnalités principales du logiciel.

Les copies d'écran des différentes fenêtres sont proposées en annexe.

4.4 Exécution des tests unitaires

Nous avons effectué deux types de tests unitaires :

- des tests fonctionnels, destinés à vérifier le bon comportement du logiciel en fonction des types de documents utilisés, des systèmes d'exploitation et des configurations logicielles diverses;
- des tests qualitatifs permettant de valider les documents transcrits.

Les tests fonctionnels ont été réalisés durant deux jours auprès d'utilisateurs expérimentés à la mission handicap de Lyon 1. La mission handicap de Lyon 1 dispose d'une grande variété d'environnements (systèmes d'exploitation, postes adaptés pour des non-voyants, matériels spécifiques...).

Le logiciel s'est chaque fois comporté de la manière souhaitée.

Les tests qualitatifs ont été réalisés à l'aide de spécialistes du braille abrégé et mathématiques. Ils nous ont permis de corriger quelques coquilles et une erreur importante dans la coupure des expressions mathématiques.

Les jeux de tests ont été réalisés à partir de jeux existants ayant servis lors de précédents tests de BraMaNet et de la mise en place du fichier d'abréviation du logiciel WinBraille par Françoise Perdoux (CNEFEI).

4.5 Étude ergonomique de l'interface

Nous avons demandé à l’équipe d’ergonomes qui participent à notre projet (Hélène DESLANDES, Vivian DUMOND et LAUTRETTE Jean-Pascal) de réaliser une étude ergonomique de l'interface.

Malheureusement, nos différents calendriers ne nous ont pas permis de prendre en compte l'ensemble des recommandations proposées, les premiers résultats nous ayant été communiqués deux jours avant la date de rendu des travaux.

Nous présentons néanmoins le travail réalisé en annexe afin de l'intégrer dans une prochaine version de NAT.

Une des difficultés d'intégration pourrait provenir du manque d'expertise des ergonomes dans le domaine spécifique du handicap, bon nombre de recommandations ne s'appliquant pas à la transcription mais plutôt aux contenus web.

Chapitre 5 : Tests utilisateurs

La phase de test s'est bien déroulée, mais s'est limitée à une utilisation experte du logiciel, faute de temps.

De plus, les testeurs avaient également participé aux premiers tests unitaires.

Une série de tests plus exhaustive est prévue pour la prochaine version du logiciel : elle se déroulera dans un premier temps au CNEFEI, puis consistera en l'analyse des retours des utilisateurs

5.1 Exécution des tests

Nous avons repris les jeux de tests précédents, augmentés de nouveaux éléments provenant d'open-office. L'annexe E présente l'ensemble des expressions mathématiques testées.

Les tests se sont déroulés à la mission handicap de Lyon 1 avec les mêmes contraintes que pour les tests unitaires, l'objectif étant cette fois-ci de valider l'ensemble du logiciel et non plus chaque fonctionnalité séparément.

5.2 Résultats de tests et validation

Les résultats ont été très encourageants. NAT s'est révélé supérieur à BraMaNet du point de vue de sa simplicité d'utilisation et de la qualité de ses transcriptions mathématiques.

En utilisation math + intégral, NAT fonctionne parfaitement.

En revanche, l'abrégé est incomplet puisque le cas général n'a pas été traité.

Les utilisateurs non-voyants préfèrent néanmoins l'utilisation de l'abrégé, même incomplet, et considèrent cette utilisation comme un plus par rapport à l'existant, les expressions abrégées ne comportent aucune erreur.

Certains éléments de l'interface ont été critiqués, et une réorganisation plus fonctionnelle des paramètres doit être envisagée.

Les testeurs nous ont encouragé à poursuivre notre travail, et comptent dès à présent utiliser NAT pour leurs transcriptions mathématiques et intégrales.

La prise en charge complète de l'abrégé est une grande attente, qui conditionne l'utilisation complète et exclusive de NAT comme transcripteur.

Plusieurs testeur nous ont assurés leur soutien pour la promotion du logiciel dès que celui-ci serait largement diffusable.

Ces résultats nous ont convaincus de l'utilité de notre logiciel, nous en poursuivrons donc le développement afin de présenter une version complète fin juin 2006.

Chapitre 6 : Communication en direction des futurs utilisateurs

6.1 Le site de téléchargement

6.1.1 Un site accessible

Le site de téléchargement est entièrement accessible et répond aux critères définis par le W3C à travers les recommandations du WAI.

Le code HTML du site fait appel à différentes feuilles de style, permettant ainsi une meilleure lisibilité pour les personnes malvoyantes. La présentation du site permet une navigation aisée à un tout utilisateur non-voyant ou malvoyant, et l'utilisation d'un navigateur textuel, de type Lynx, permet d'ailleurs de vérifier que la totalité de l'information visualisable à l'écran est aussi disponible en mode texte, pour un affichage sur une plage braille ou une sortie sur une synthèse sonore.

Le document ne comporte que deux images graphiques (logos Lynx et W3C) possédant une fonction alt qui indique clairement le contenu de ceux-ci.

6.1.2 La navigation de page

Un menu accessible est placé en haut de page et chaque rubrique est clairement identifiée par son nom. Chaque fin de rubrique comporte un retour vers le haut de page et donc un retour vers l'accès au menu général de la page pour une meilleure accessibilité.

6.1.3 Les différentes rubriques du menu accessible

Le menu général comporte les huit rubriques suivantes :

- **NAT version 1.0 - GNU General Public License (GPL)**

  Descriptif sommaire du logiciel et de la population à laquelle il s'adresse.

- **Pour télécharger gratuitement NAT**

  Lien pour télécharger le programme d'installation du logiciel NAT.

- **Comment installer NAT ?**

  Quelques recommandations pour l'installation.

- **Consulter le guide utilisateur**

  Guide utlisateur en ligne et au format *texte*.

- **Programmes nécessaires à NAT**

  Différents liens pour télécharger les programmes utiles au bon fonctionnement du logiciel NAT.

- **Télécharger les sources de NAT**

NAT est un logiciel GNU General Public License (GPL), ses sources sont donc mises à disposition et donc téléchargeable à partir de cette rubrique.

- **L'origine de NAT**

  Résumé sommaire de l'origine du projet NAT.

- **Contactez-nous**

  Pour toutes demandes d'informations, questions ou remarques concernant l'utilisation du logiciel, les utilisateurs peuvent nous contacter à travers par le biais de nos adresses électroniques.

6.2 La plaquette de présentation

NAT 1.0

Transcripteur universel de documents standards en texte braille

http://nzbb.free.fr/NAT

Qu'est-ce que NAT ?

NAT est un logiciel libre de transcription automatique capable de convertir plusieurs formats de fichiers en braille. Il gère les documents textuels, littéraires ainsi que de tout document contenant des expressions mathématiques (pour la version 1.0). Les versions suivantes incluront d'autres écritures (musique, chimie, etc.).

Il permet de transcrire les documents selon les règles de présentation du braille, en abrégé ou en intégral. NAT version 1.0 offre à toute personne la possibilité de produire une transcription de qualité sans posséder de connaissances en braille. Les symboles et expressions mathématiques actuellement implémentées vont jusqu'au niveau universitaire.

NAT permet ainsi à un professeur de mathématiques de fournir, à un élève non-voyant intégré dans sa classe, le même document, mais transcrit, que celui destinés aux élèves voyants.

Télécharger NAT 1.0 « Master Handi »

Vous pouvez télécharger NAT version 1.0 directement sur le site de NAT.

Comment installer NAT ?

Après avoir dézippé le fichier NAT-v1-0.zip , exécutez le programme NAT.bat et suivez les instructions. Si le programme affiche des erreurs comme des librairies ou des classes non trouvées, il vous manque peut-être une machine virtuelle JAVA. Reportez vous à la rubrique **'Ressources nécessaires'** que vous trouverez ci-dessous.

Consulter le guide utilisateur

Pour en savoir plus, et avant toute utilisation, reportez vous au guide de l'utilisateur en ligne (format texte),

Ressources nécessaires à NAT

NAT nécessite environ 5 Mo libres d'espace disque et fonctionne dans tout environnement (systèmes Linux, Microsoft Windows version 98, ME, NT, 2000, ou XP, Mac OS) disposant d'une machine virtuelle JAVA 1.5 ou supérieure (http://java.sun.com/j2se/1.5.0/download.jsp , Java Runtime Environment JRE 5.0).

Télécharger les sources de NAT

NAT est un logiciel libre sous licence GNU General Public License 2 (GPL 2) (http://www.gnu.org/copyleft/gpl.html) : Vous pouvez télécharger directement les sources du logiciel sur le site de NAT.

L'origine de NAT

Le logiciel NAT a vu le jour dans le cadre d'un projet technique réalisé par deux étudiants, Bruno Blanchard et Bruno Mascret, du Master « Technologie et Handicap» de l'université Paris 8 (http://ufr6.univ-paris8.fr/desshandi/index.php). Il prolonge et améliore le logiciel Bramanet (http://handy.univ-lyon1.fr/projets/bramanet/), développé par la mission handicap de l'université Lyon 1.

Remerciements

Nos remerciements s'adressent tout particulièrement à Frédéric Schwebel et ses collaborateurs (projet BraMaNet, Mission Handicap de Lyon 1), Henrik Just (Writer2Latex), la communauté des développeurs d'OpenOffice et Benoît Dasset (fonte BrailleAntoine), pour les éléments de programmation intégrés. Ils s'adressent également à Jaime Lopez Krahé (directeur du master Handi) et Pascale Pousset (gestion du projet master Handi), pour leur mission d'encadrement du projet. Nous remercions aussi Marc Ollier (CNEFEI de Suresnes), pour ses conseils, Hélène Deslandes, Vivian Dumont et Jean Pascal Lautrette (étudiants du Master d'ergonomie Paris 8), pour leur participation à l'étude de l'ergonomie de l'interface, et la Mission Handicap de Lyon 1, pour la réalisation des tests. Enfin, nos plus vifs remerciements s'adressent aux familles Blanchard et Mascret, Aude Lancelle et Isabelle Blanchard, pour leur infinie patience et leur soutien moral tout au long du projet.

Contactez-nous

Envoyer vos remarques ou questions à bmascret@free.fr ou brblanchard@wanadoo.fr

Conclusion

Les progrès technologiques sont en train de bouleverser totalement le traitement de l'information. Nous nous orientons progressivement vers une production de documents électroniques, qui restent aujourd'hui le support de prédilection de la transmission de l'information en général et de l'éducation en particulier.

Toutefois, il est indispensable de préparer les solutions qui permettront dans un avenir proche de délivrer des documents multimédias adaptés aux caractéristiques de chaque utilisateur.

Le développement du présent projet permet d'ouvrir de nouvelles perspectives d'accessibilité de l'information scientifique et littéraire aux personnes non voyantes, ce qui permet notamment l'amélioration de l'intégration de cette population.

Nous pensons par notre modeste travail avoir contribué à ces orientations et démontré la faisabilité d'un transcripteur universel en braille.

Les encouragements des utilisateurs nous ont convaincus de l'utilité de ce logiciel et de son adéquation avec leurs besoins.

Néanmoins, NAT nécessite des améliorations à court terme que ce soit au niveau du confort d'utilisation (interface graphique ergonomique) ou de ses performances quant au braille abrégé. La prochaine version intégrera ces éléments.

A plus long terme, NAT devrait offrir une grande variété de choix de transcriptions (chimie, musique, langues étrangères). NAT pouvant s'intégrer facilement dans d'autres programmes (car exécutable en ligne de commande), de nombreuses extensions sont envisageables comme par exemple la transcription à la volée de texte brut par un lecteur d'écran.

Références bibliographiques

[BELL01] Z.Bellahcene, X.Baril, « XML et les systèmes d'intégration de données», Ingénierie des systèmes d'information, vol. 6, n°3, p11-32, 2001.

[BERN00] J.C.Bernadac, « Construire une application XML : avec deux études de cas réels » Editions Eyrolles, 2000.

[KAY01] M.Kay, « XSLT 2ème édition » Edition CampusPress, 2001.

[MARC00] B.Marchal, « XML » Edition CampusPress, 2000.

[MICH01] A.Michard, « XML langage et applications» Editions Eyrolles, 2001.

[MULL00] P.A.Muller, N.Gaertner, « Modélisation objet avec UML» 2ème édition, Edition Eyrolles, 2000.

[YOUN00] M.J.Young, «Formation à XML» Edition Microsoft Press, 2000

Sites Web référencés

[WEB01]. W3C (World Wide Web Consortium)

http://www.w3.org/

[WEB02] WAI (Web Accessibility Initiative)

http://www.w3.org/WAI/

[WEB03]. HTML (Hyper Text Markup Language)

http://www.w3.org/MarkUp/

[WEB04]. XML (Extensible Markup Language)

http://www.w3.org/XML/

http://www.educnet.education.fr/dossier/xml/default.htm

http://www.w3.org/XML/Schema

http://zuse.esnig.cifom.ch/intranet/SpecSchema/xschema.htm

[WEB05]. MathML (Mathematical Markup Language)

http://www.w3.org/Math/

[WEB06]. XSL (Extensible Stylesheet Language)

http://www.w3.org/Style/XSL/

[WEB07] Le projet " Bramanet", mission Handicap de l'université Lyon 1

http://handy.univ-lyon1.fr/projets/bramanet/

[WEB08] Un traducteur de mathématiques de l'université Paris 6

http://inova.snv.jussieu.fr/maths/

[WEB09] Access to Mathematics for Blind Students, Arthur I. KARSHMER, University of South Florida, USA, Journée thématique BrailleNet 2002, Cité des Sciences et de l'Industrie de La Villette

http://www.snv.jussieu.fr/inova/villette2002/res5.htm

[WEB10] Code pour la transcription en Braille de la notation Mathématique, Comité de normalisation du braille français en éducation Édition révisée, Mai 2001

http://www.meq.gouv.qc.ca/ens-sup/ens-coll/braille/Code_braille_mathematique.pdf

[WEB11] Le site de l'Association Valentin Haüy, l'AVH

http://www.avh.asso.fr/

[WEB12] Le site de l'Institut National des Jeunes Aveugles, l'INJA

http://www.inja.fr

[WEB13] Le site de l'Association des Professeurs de Mathématique de l'Enseignement Public, l'APMEP

http://www.apmep.asso.fr/

[WEB14] Le site de l'IDE Eclipse

[WEB15] Le site de l'éditeur Kate

[WEB16] Le site de la distribution Debian de Linux

http://www.debian.org

Glossaire

Afficheur Braille

Ceci désigne un terminal connecté à l'ordinateur et restituant l'information qu'il reçoit sous forme de texte en Braille. Synonyme(s): terminal Braille, plage tactile, plage Braille.

Agrandissement de l'écran (logiciel)

Logiciel qui permet aux utilisateurs ayant une basse vision d'agrandir l'image sur l'écran autant de fois qu'ils le souhaitent. Certains programmes agrandissent tout l'écran, d'autres, seulement l'espace qui entoure le curseur de la souris de sorte que la portion agrandie bouge selon les déplacements du curseur. Les fonctions d'accessibilité peuvent être intégrées au système d'exploitation, notamment les fonctions pour agrandir l'écran, choisir le modèle et la taille de la souris, et contrôler le contraste des couleurs. Marques de produits : ***Large Print for Windows*** de Visionware Software, ***MAGic Screen Magnification*** de Henter-Joyce, ou ***Zoom Text Xtra*** de Al Sqaured.

Appareil de prise de notes en braille

C'est un appareil électronique associant un clavier braille, de la mémoire et une plage tactile de lecture. L'élève aveugle peut taper silencieusement ses cours directement en braille et contrôler la frappe en effleurant des doigts sa plage tactile.

Assistant numérique

Ordinateur portable conçu pour fonctionner de la même façon qu'un agenda électronique personnel, un appareil de prise de notes et/ou différents dispositifs de communication. Il existe aussi un groupe de PDA conçus pour ceux et celles ayant un handicap. Ces PDA utilisent la transmission sonore des données, l'affichage en braille et le clavier braille pour intégrer l'interface-utilisateur.

Braille

Système notationnel tactile qui permet de former des combinaisons de points en relief correspondant aux lettres de l'alphabet, aux chiffres et aux signes de ponctuation, et qu'on peut lire du bout des doigts.

Braille intégral ou braille abrégé

Louis Braille inventa vers 1828 un système de représentation des caractères d'imprimerie " noirs " lus avec les yeux par des combinaisons de points saillants (de 1 à 6 points) embossés sur du carton lus tactilement avec les doigts. Cela constitue le braille intégral qui fait correspondre à chaque lettre ordinaire une matrice de points embossés. Le braille intégral est un code simple très puissant mais qui prend beaucoup de place. C'est pourquoi, après quelques années d'apprentissage et d'utilisation pour appréhender l'orthographe, on enseigne aux jeunes aveugles une forme condensée du braille, c'est le braille abrégé qui réduit d'environ 1/3 le nombre de caractères braille et qui donne donc plus d'informations pour un même volume de données. On considère généralement que la connaissance du braille abrégé est indispensable à la poursuite des études dans le secondaire et à l'université.

Braille abrégé de type II

Les mots et le texte sont abrégés afin d'augmenter la vitesse de lecture et de réduire l'espace requis pour la transcription.

Braille non abrégé de type I

Reproduction caractère par caractère d'un texte imprimé en diverses combinaisons de points.

Calculatrice avec affichage de grande dimension

Calculatrice scientifique qui affiche en rouge des caractères numériques d'un demi-pouce pour les malvoyants.

Calculatrice vocale

Calculatrice qui exécute les calculs mathématiques habituels et communique les résultats par l'intermédiaire d'un dispositif de synthèse vocale.

Commande vocale

C'est la possibilité d'entrer sur l'ordinateur des commandes par la voix, c'est aussi la dictée vocale.

Convertisseur en braille

Appareil utilisé comme une machine à écrire classique qui tape les caractères en braille sur du papier, par exemple le convertisseur Perkins., SmartView.

Dictionnaire Franklin ("*Franklin Speller*")

Dictionnaire « sonore » avec affichage simultané de la définition, des syllabes et de la nature grammaticale de chaque mot. Utile pour les personnes qui ont une cécité totale ou une basse vision ou qui ont des troubles d'apprentissage.

Dispositifs haptiques

Une interface haptique est un dispositif qui permet une communication interactive entre l'utilisateur, l'utilisatrice et l'ordinateur en générant des stimulations tactiles lors de l'exploration d'une image. Il existe deux types importants de dispositifs haptiques : 1) un gant ou un stylet tactile qui permet à l'utilisateur de « toucher » et de manipuler des objectifs virtuels tridimensionnels; 2) un dispositif qui permet à l'utilisateur ou à l'utilisatrice de « sentir » la texture d'objets bidimensionnels à l'aide d'une interface qui ressemble à un stylo ou à une souris d'ordinateur.

Embosseuse

C'est une sorte d'imprimante qui au lieu d'imprimer des caractères d'imprimerie, édite un texte en caractères braille sur une feuille de papier cartonné sous la forme de milliers de petites bosses. Le texte embossé est lu tactilement par l'enfant aveugle.

Gros caractères

Lettres ou chiffres agrandis jusqu'à 150 fois plus que la normale à l'aide de différents appareils, par exemple un photocopieur ou un logiciel de traitement de texte. On utilise habituellement une police de caractères de 16 ou de 18 points, mais elle peut être plus grande encore, selon les besoins de la personne.

Gros caractères (logiciel)

Il s'agit d'un logiciel qui agrandit certaines parties (ou la totalité) de l'écran d'un ordinateur, soit jusqu'à 16 fois plus que la normale. Il transforme le dispositif de pointage ou le curseur en un outil qui agit comme une loupe sur la page imprimée.

Horloges ou montres parlantes

Ces appareils disent l'heure lorsqu'on appuie sur un bouton.

Imagerie tactile

Production de lignes ou de points en relief qui peuvent être touchés et interprétés par les utilisateurs non-voyants.

Imprimante en braille (dispositif d'embossage du braille)

Il s'agit d'une imprimante générant du braille (points tactiles sur le papier) plutôt qu'un texte.

Intervenant pour les personnes ayant une surdi-cécité

Personne formée pour procurer un service d'interprétation simultanée ainsi que d'autres services de communication et des renseignements connexes à une personne ayant une surdi-cécité. Selon la méthode qui convient le mieux, l'intervenant peut communiquer par le langage visuel des signes, le langage gestuel des signes, l'épellation digitale, le braille ou des notes rédigées en gros caractères (si la personne a une basse vision).

Lecteur

La tâche d'un lecteur est de lire le contenu d'un texte à un candidat ou à une candidate qui a une cécité totale ou une basse vision ou qui a des difficultés sur le plan du traitement des images. Le lecteur ou la lectrice lit les différentes réponses possibles s'il s'agit d'un examen à choix multiples. Cette personne doit bien lire et bien articuler. Il est bon que le lecteur ou la lectrice se familiarise avec les procédures et la terminologie utilisée avant l'examen, surtout si le contenu est de nature technique.

Lecteur optique de caractères (logiciel)

Dispositif qui convertit l'image d'un texte, par exemple un document balayé par scanner ou un fichier transmis par télécopieur, en renseignements selon une forme utilisée par l'ordinateur. On ne peut pas modifier le texte dans une image : les lettres sont faites de petits points (pixels) qui, ensemble, forment l'image du texte. Le lecteur optique de caractères numérise une image et convertit les lettres illustrées en un texte modifiable selon la forme des pixels dans l'image. Une fois la lecture optique terminée, on

peut exporter le texte converti et l'utiliser dans une variété d'applications pour le traitement de texte, la mise en page et la création de chiffriers. La lecture optique de caractères permet aussi aux lecteurs d'écran et aux affichages régénérés en braille de lire le texte contenu dans les images.

Lecteur sonore d'écran

La technologie de sortie vocale. L'équipement et le logiciel détectent et transforment le texte qui apparaît à l'écran de l'ordinateur ainsi que les lettres tapées sur le clavier, puis les envoient à un synthétiseur de la parole. Marques de produits : JAWS for Windows de Henter-Joyce Inc., OutSpoken for Macintosh de ALVA Access Group Inc., Screen Reader 2 de IBM.

Linéariser :

La linéarisation d'une page consiste à présenter son contenu dans l'ordre dans lequel il est disposé dans le code source. Le résultat peut alors s'avérer différent de celui obtenu sous un navigateur graphique, en particulier concernant les tableaux.

Logiciel texte-braille

Convertit le texte en format braille afin qu'il puisse être enregistré, imprimé ou transféré à un écran tactile. La plupart des traducteurs montrent également une présentation visuelle des points à l'écran. La plupart des programmes permettent la traduction vers le braille abrégé, le braille non abrégé ou les deux versions de braille.

Loupe

Outil portatif qui offre une vue agrandie de ce qui se trouve en-dessous.

Machine Perkins braille

C’est la machine mécanique qui permet de taper du braille sur une feuille de papier cartonné. C’est un équipement très robuste mais un peu volumineux et surtout très bruyant et donc peu commode dans le cadre d’une intégration.

Médias substituts

Document imprimé accessible sous une forme non traditionnelle (braille, gros caractères, audio-cassette et version électronique) à l'intention des personnes ayant une basse vision ou des troubles d'apprentissage.

Moniteur à grand écran

Moniteur dépassant 19 pouces; pratique pour les personnes ayant une basse vision.

Navigateur graphique

Navigateur Web qui permet la visualisation à l'écran d'éléments graphiques, et notamment d'images. Synonyme(s): navigateur en mode graphique.

Navigateur textuel

Ce sont des navigateurs par lesquels seuls les éléments textuels d'une page Internet sont restitués à l'utilisateur. Synonyme(s): navigateur en mode texte.

Navigateur vocal

Navigateur Internet doté d'une synthèse vocale intégrée et de fonctions supplémentaires permettant d'agir sur la manière dont l'information est prononcée.

Plage tactile

C'est une barrette électronique de 20, 40 ou 80 caractères braille générés généralement par des petites cellules piézo-électrique faisant monter ou descendre des petits picots de plastique formant les caractères braille. Le texte braille est éphémère: il apparaît et disparaît en un flux de caractères que la personne aveugle peut contrôler et lire tactilement.

Preneur de notes en braille

Dispositif électronique qui permet de prendre des notes à l'aide d'un clavier braille. Ces tablettes d'écriture portatives sont pratiques dans le cadre de réunions, etc.

Preneur de notes portatif

Dispositif électronique qui permet de prendre des notes en braille; utilisés dans les réunions, etc. Il peut être branché à un ordinateur en vue d'imprimer l'information notée.

Scanner

Dispositif permettant de lire un texte ou des illustrations imprimées et de les convertir en renseignements selon une forme utilisée par l'ordinateur. Les scanners optiques ne distinguent pas un texte d'une illustration; ils représentent toutes les images comme étant en mode point. Pour éditer un texte lu au moyen d'un scanner optique, un lecteur optique de caractères est nécessaire afin de convertir l'image en caractères ASCII. La plupart des scanners optiques sont vendus avec des ensembles de lecteurs optiques de caractères.

Synthétiseur texte-parole

Système qui permet de convertir un document informatique (p. ex. un document produit à l'aide d'un logiciel de traitement de texte ou une page Web) en symboles phonétiques audibles par l'entremise du haut-parleur de l'ordinateur. Utile pour les personnes qui ne peuvent pas voir ou lire un texte affiché sur l'écran de l'ordinateur.

Synthèse vocale

Dispositif logiciel qui transforme une information sous forme de voix. Les lecteurs d'écran et navigateurs vocaux sont généralement livrés avec une ou deux synthèses vocales intégrées et capables de traiter plusieurs langues. Synonyme(s): voix de synthèse, moteur de synthèse vocale.

Synthèse de voix ou synthèse vocale C'est la possibilité qu'offre l'ordinateur de synthétiser de la parole à partir d'une suite de caractères écrits et mémorisés.

Synthétiseur vocal externe et portable

Synthétiseur vocal portable que l'utilisateur ou l'utilisatrice peut transporter facilement et brancher à un autre ordinateur, par exemple, lorsqu'il voyage pour affaires. (Un synthétiseur est un logiciel qui convertit des renseignements textuels codés ou d'autres informations électroniques et les reproduit de manière orale).

Système de reconnaissance vocale

Technologie qui permet à celui ou à celle qui l'utilise de dicter un texte à l'ordinateur ou de lui adresser des commandes verbales (par exemple pour ouvrir les programmes de diverses applications, faire dérouler des menus ou sauvegarder des documents). Bien qu'on ait amélioré l'exactitude des systèmes de reconnaissance vocale au cours des dernières années, ils ne sont pas encore parfaits et ne fonctionnent bien que dans des circonstances particulières pour des utilisatrices et des utilisateurs particuliers. Marques de produits : Dragon Dictate, Naturally Speaking de Dragon Systems, Freespeech de Philips, Via Voice de IBM, Voice Pilot for Windows, ou Dragon Naturally Mobile.

Tablette braille

C'est une petite tablette associée à une réglette perforée qui permet d'écrire à la main du braille à l'aide d'un poinçon. C'est un matériel suffisamment réduit pour être contenu dans la poche de l'élève.

Télé-agrandisseur

C'est un appareil qui permet d'agrandir sur un écran une image ou un texte placé sous la lentille d'un zoom réglable.

Technologie d'adaptation

Programmes et logiciels informatiques mis à la disposition des personnes qui ont de la difficulté à accéder aux systèmes d'information en utilisant les méthodes traditionnelles (de plus en plus souvent appelée « technologie habilitante » ou « technologie fonctionnelle »).

Télévision en circuit fermé (TCF)

Il s'agit d'un système de grossissement de l'image constitué d'un écran vidéo couplé à une caméra vidéo. Cette technologie a d'importants avantages pour les personnes ayant une basse vision. La TCF autonome peut être reliée à un téléviseur, à un moniteur vidéo ou à un écran d'ordinateur. L'utilisateur, l'utilisatrice peut contrôler la luminosité et le contraste, inverser l'avant-plan et l'arrière-plan et régler la gamme des gris.

W3C (World Wide Web Consortium)

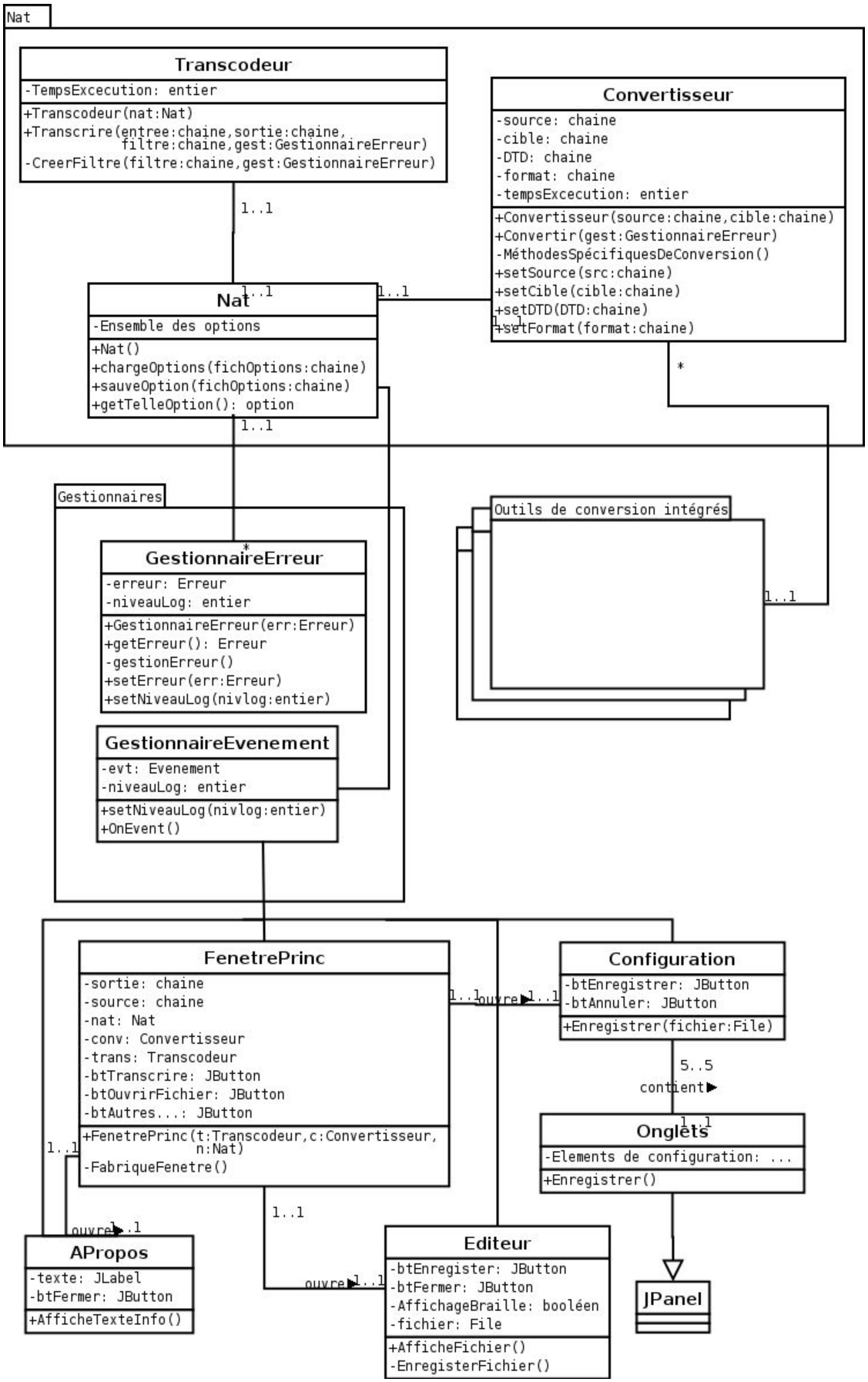
Le World Wide Web Consortium (W3C) a pour mission de mener le World Wide Web au maximum de son potentiel en développant des protocoles communs qui encouragent son évolution et assurent son interopérabilité.

WAI

Le Web Accessibility Initiative a été créé en 1999. C'est un département du w3c, qui établit des recommandations internationales pour rendre les sites Internet accessibles à toutes personnes.

WCAG (Web Content Accessibility Guidelines) Directives pour l'accessibilité aux contenus WEB est un référentiel international publié par le WAI, branche du w3c. Il définit des recommandations et des techniques pour produire des sites internet accessible à tous. La version actuellement en vigueur est WCAG1.0.

Annexe A : Modèle général



Annexe B : Code braille informatique 8 points français pour Windows

(Table de référence FRANÇAISCP-1252 ou FRANÇAIS)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	0000	0016	0032	0048 0	0064 @	0080 P	0096 `	0112 p	0128 €	0144	0160	0176 °	0192 À	0208 Ð	0224 à	0240 ð
1	0001	0017	0033 !	0049 1	0065 A	0081 Q	0097 a	0113 q	0129	0145 ‘	0161 ¡	0177 ±	0193 Á	0209 Ñ	0225 á	0241 ñ
2	0002	0018	0034 "	0050 2	0066 B	0082 R	0098 b	0114 r	0130 ‚	0146 ’	0162 ¢	0178 ²	0194 Â	0210 Ò	0226 â	0242 ò
3	0003	0019	0035 #	0051 3	0067 C	0083 S	0099 c	0115 s	0131 *f*	0147 “	0163 £	0179 ³	0195 Ã	0211 Ó	0227 ã	0243 ó
4	0004	0020	0036 $	0052 4	0068 D	0084 T	0100 d	0116 t	0132 „	0148 ”	0164 ¤	0180 ´	0196 Ä	0212 Ô	0228 ä	0244 ô
5	0005	0021	0037 %	0053 5	0069 E	0085 U	0101 e	0117 u	0133 …	0149 •	0165 ¥	0181 µ	0197 Å	0213 Õ	0229 å	0245 õ
6	0006	0022	0038 &	0054 6	0070 F	0086 V	0102 f	0118 v	0134 †	0150 –	0166 ¦	0182 ¶	0198 Æ	0214 Ö	0230 æ	0246 ö
7	0007	0023	0039 '	0055 7	0071 G	0087 W	0103 g	0119 w	0135 ‡	0151 —	0167 §	0183 ·	0199 Ç	0215 ×	0231 ç	0247 ÷
8	0008	0024	0040 (	0056 8	0072 H	0088 X	0104 h	0120 x	0136 ˆ	0152 ˜	0168 ¨	0184 ¸	0200 È	0216 Ø	0232 è	0248 ø
9	0009	0025	0041)	0057 9	0073 I	0089 Y	0105 i	0121 y	0137 ‰	0153 ™	0169 ©	0185 ¹	0201 É	0217 Ù	0233 é	0249 ù
A	0010	0026	0042 *	0058 :	0074 J	0090 Z	0106 j	0122 z	0138 Š	0154 š	0170 ª	0186 º	0202 Ê	0218 Ú	0234 ê	0250 ú
B	0011	0027	0043 +	0059 ;	0075 K	0091 [	0107 k	0123 {	0139 ‹	0155 ›	0171 «	0187 »	0203 Ë	0219 Û	0235 ë	0251 û
C	0012	0028	0044 ,	0060 <	0076 L	0092 \	0108 l	0124 ¦	0140 Œ	0156 œ	0172 ¬	0188 ¼	0204 Ì	0220 Ü	0236 ì	0252 ü
D	0013	0029	0045 -	0061 =	0077 M	0093]	0109 m	0125 }	0141	0157	0173 -	0189 ½	0205 Í	0221 Ý	0237 í	0253 ý
E	0014	0030	0046 .	0062 >	0078 N	0094 ^	0110 n	0126 ~	0142 Ž	0158 ž	0174 ®	0190 ¾	0206 Î	0222 Þ	0238 î	0254 þ
F	0015	0031	0047 /	0063 ?	0079 O	0095 _	0111 o	0127	0143	0159 Ÿ	0175 ¯	0191 ¿	0207 Ï	0223 ß	0239 ï	0255 ÿ

Annexe C : Les instructions XSLT

Cette annexe présente une liste alphabétique des entrées de référence de chacun des éléments XSLT.

INSTRUCTION	DÉSCRIPTION
xsl:apply-imports	Utilisée de paire avec les feuilles de style importées.
xsl: apply-templates	Définit un ensemble de nœuds à traiter et obtient que le système les traite en sélectionnant une règle modèle appropriée pour chacun de ces nœuds.
xsl:attribute	Affiche un nom et une valeur d'attribut dans la sortie courante.
xsl:attribute-set	Définit un ensemble nommé de noms et de valeurs d'attributs.
xsl:call-templete	Utilisée pour appeler un modèle nommé. Son effet est analogue à un appel de procedure dans les autres langages de programmation.
xsl:choose	Definit un choix entre un certain nombre d'alternatives.
xsl:comment	Sert à écrire un commentaire vers la sortie courante.
xsl:copy	Copie le nœud courant du document source dans la sortie courante.
xsl:copy-of	Copie un ensemble de nœuds vers la sortie courante.
xsl:decimal-format	Sert à définir les caractères et les symboles utilisés lors de la conversion de nombre en chaînes à l'aide de la fonction **format-number()**
xsl:document	Sert à créer un nouveau fichier de sortie.
xsl:element	Sert à afficher un nœud élément vers la sortie courante.
xsl:fallback	Sert à définir le traitement qui doit avoir lieu en cas d'indisponibilité de l'instruction définie par son parent.
xsl:for-each	Sélectionne un ensemble de nœuds à l'aide d'une expression XPATH et effectue le même traitement pour chaque nœud de l'ensemble.
xsl:if	Comprend un corps de modèle qui est instancié uniquement si une condition spécifier est vrai.
xsl:import	Permet d'importer le contenue d'un module de feuille de style dans un autre.
xsl:include	Utilisé pour inclure le contenue d'un module de feuille de style dans un autre.
xsl:key	Sert à déclarer une clé nommé a utiliser avec la fonction **key()** dans des expressions et des motifs.
xsl:message	Affiche un message et met fin de manière facultative à l'exécution de la feuille de style.
xsl:namespace-alias	Permet de mettre en correspondance un espace de noms utilisé dans la feuille de style vers un autre espace de noms utilisé dans la sortie.
xsl:number	Sert à allouer un numéro séquentiel au nœud courant et à mettre en forme un nombre à afficher.
xsl:otherwise	Est utilisé dans une instruction **<xsl :choose>** afin d'indiquer l'action à entreprendre lorsque aucune des conditions **<xsl :when>** n'est remplie.
xsl:output	Permet de contrôler le format de la sortie de la feuille de style.

xsl:param	Utilisé pour décrire un paramètre global qu'immédiatement dans un élément **<xsl :template>** pour décrire un paramètre locale dans un modèle.
xsl:preserve-space	Permet de contrôler le mode de gestion des nœuds espaces dans le document source.
xsl:processing-instruction	Permet de décrire un nœud instruction de traitement dans la sortie courante.
xsl:script	Permet de définir des fonctions d'extension qui peuvent être appelées à partir d »expression XPATH.
xsl:sort	Permet de définir une clé de tri, pour spécifier l'ordre dans lequel les nœuds sélectionnés par **<xsl :apply-tempates>** ou **<xsl :for-each>**
xsl:strip-space	Equivalent à **< xsl:preserve -space>**
xsl:stylesheet	Est l'élément le plus à l'extérieur d'une feuille de style.
xsl:template	Définit un modèle permettant de produire un résultat.
xsl:text	Utilisé dans un corps de modèle pour afficher le texte littéral dans la sortie courante.
xsl:transform	Equivalent au **<xsl :stylesheet>**.
xsl:value-of	A pour effet de d'écrire la valeur textuelle d'une expression dans l'arbre résultat.
xsl:variable	Permet de déclarer une variable locale ou globale dans une feuille de style et de lui attribuer une valeur.
xsl:when	Il apparaît toujours comme enfant de l'élément **<xsl :choose>**. Il définit une condition à tester et l'action à entreprendre si la condition est vraie.
xsl:with-param	Permet de définir les valeurs des paramètres lors de l'appel d'un modèle à l'aide de **<xsl : call-template>** ou de **<xsl :apply-templates>**.

Annexe D : Étude ergonomique d'une interface graphique

Avertissement : Cette annexe a été réalisée par l'équipe d'ergonomes suite à nos travaux communs.

D.1 Demande initiale

Réaliser une interface utilisateur ergonomique permettant de la paramétrer et de l'utiliser aussi bien par des utilisateurs voyants ou non-voyants. Cette interface devra être à la fois facilement utilisable par un utilisateur novice, et par un utilisateur plus expérimenté qui pourra contrôler toutes les étapes du processus de transcription.

D.2 Reformulation

Le premier travail des concepteurs est l'élaboration et la mise en œuvre des codes de traduction. Dans le processus de conception, l'élaboration de l'interface est prévue à la fin du développement. Afin de favoriser une action ergonomique satisfaisante, il est important de s'inscrire au sein de l'évolution de la conception du logiciel. La demande des concepteurs est l'apport de recommandations ergonomiques pour l'élaboration de l'interface. Pour favoriser une interface adéquate, il est important de connaître les caractéristiques des utilisateurs visés. Ces utilisateurs peuvent être voyants, malvoyants ou non-voyants. (Un focus détaillé de la population utilisatrice semble indispensable comprenant : le niveau d'expertise, le matériel utilisé…)

A partir de ces informations, il sera possible d'établir les grandes lignes de l'interface possible : interface universelle versus interface voyants et interface non-voyants. L'analyse des caractéristiques des futurs utilisateurs et de leurs propositions permettra également d'isoler des menus utiles à chaque type de population. La prise en compte de ces différentes caractéristiques permettra d'aboutir à la mise en avant des critères :

- D'utilité : adéquation entre buts d'un utilisateur et les finalités du logiciel.
- D'utilisabilité : facilité avec laquelle un utilisateur peut mettre en œuvre un logiciel dans la réalisation
- D'utilisation : compatibilité entre utilité et utilisabilité.

A la suite de ce travail préparatoire, il sera possible d'établir des critères généraux permettant une utilisation facile et intuitive du logiciel en fonction des caractéristiques de la ou des futures populations utilisatrices.

D.3 Organisation de l'étude

- Recueil d'information auprès des concepteurs, ayant déjà conçu un logiciel transcripteur de mathématiques en braille.
- Analyse de la demande des concepteurs et du contenu à intégrer dans le logiciel.
- Etablir une liste de critères pour les interfaces voyants et non voyants.
- Faire une comparaison de ces critères et réfléchir à la faisabilité d'un logiciel commun ou non et communiquer ces recommandations aux concepteurs, pour qu'ils puissent développer une maquette.
- Suite à l'analyse de la demande et à quelques entretiens par mail, établir un questionnaire pour cerner la population, et mieux connaître l'utilisation relative à ces critères qui est faite du logiciel.
- Diffuser le questionnaire auprès des futurs utilisateurs et exploiter les résultats.
- Comparer ces résultats aux critères d'ergonomie a priori, en gardant bien en tête l'acceptabilité (par exemple : « le fait que les voyants préfèrent utiliser des logiciels pour les voyants même si leur utilisation est moins adaptée ».)

La modélisation des maquettes pour les tests utilisateurs sera faite d'après les recommandations ergonomiques et selon trois modalités :

- La première maquette à partir d'une base commune d'après les critères d'une interface en commun.
- La deuxième sera personnalisée pour les non-voyants.
- Et la troisième pour les voyants.

Grâce à la capture d'écran en vidéo, on pourra mesurer le nombre d'erreurs et le temps mit pour réaliser un parcours standard et ainsi ce faire une idée de l'efficacité de chacune des modalités suivant les critères utilisateurs. On pourra par ailleurs lier un questionnaire de satisfaction pour comprendre si l'interface la plus efficace est aussi la plus appréciée. Les résultats de ces tests amèneront à développer une deuxième série de recommandations.

D.4 Critères généraux ergonomiques pour les interfaces voyants

Le premier critère, lié à **l'organisation visuelle** comporte différents points :

- La structuration de l'information, avec diverses recommandations liées à la hiérarchie visuelle,
- La longueur des pages : liée à l'accessibilité et au caractère déroutant.
- La construction de la page, ou l'on évoque sa lisibilité

Le deuxième critère est lié à la **page d'accueil**, elle doit être allégée, informative et jouer le rôle de guide.

Le troisième critère, est lié au **graphisme**, le bruit visuel doit être limité pour que le texte soit lisible. L'insertion d'image doit être utile et elles doivent pouvoir être décrites grâce à l'attribut « alt », les mêmes critères s'appliquent aux animations.

Le quatrième critère est lié au **texte**, les éléments clés doivent être mis en saillance, l'écriture Web doit être concis et concrète, la taille des caractères doit être modifiable pour la lisibilité des caractères.

Le cinquième critère concerne **les cases** de types formulaire, la valeur par défaut doit être la plus courante, l'utilisateur doit être guidé, et les boutons seront préférés à la liste de sélection.

*A destination de la navigation entre ces pages :

Le premier critère, est lié au **système de navigation**, il est pertinent de mentionner où se trouve l'utilisateur, par où il est passé, de fournir une vue globale et de limiter la complexité avant chaque clic.

Le deuxième critère est lié aux **items de navigation**. Il faut prévoir des barres de navigation s'il y a plusieurs pages, éviter les sous menus, les listes de sélections ne doivent pas être utilisées pour la navigation, mais uniquement pour les formulaires, et enfin éviter les boutons d navigations sans texte explicatif.

Le troisième critère est lié au **chemin de progression**, il faut toujours indiqué le titre de la page courante dans le chemin de progression même si il y a redondance, cela permet de se repérer.

Le quatrième critère, est lié aux **liens**, il faut le placer en fin de phrase pour ne pas interrompre la lecture, doit permettre à l'internaute de prévoir le contenu. Il faut par ailleurs placer les liens sur les mots clés et évité de placer les liens externes sur une nouvelle fenêtre, car souvent perçu comme pop up.

D.5 Critères pour les interfaces non voyants

Quatre grands critères sont utiles pour la conception de logiciels non voyants :

- Tout d'abord, le logiciel doit être **perceptible**, c'est-à-dire que le contenu doit être perçu par l'ensemble des utilisateurs. Ainsi les éléments utiles pour les voyants, tels que l'identification par des indices visuels des liens hypertextes s'avèrent inutile pour les non-voyants. De même, les images ne servent à rien sans leur équivalent textuel.
- Ensuite, il doit être **utilisable**, c'est-à-dire qu'une personne incapable d'utiliser une souris doit être en mesure de se déplacer dans le contenu. L'exploration pas à pas nécessite un contenu structuré, grâce notamment à des en-têtes. Par ailleurs, les champs avec formulaire doivent être précisé par des étiquettes pour que le logiciel de lecture de l'écran soit efficace.
- Le troisième point est le critère de **compréhension**, un langage simple et un indicateur de changement de langue pour rendre la traduction efficace, sont à prévoir. S'assurer que chaque lien à un sens quand il est lu seul, et structurer les pages avec un en tête.
- Le dernier critère est la **robustesse**, qui permet de travailler avec des technologies adaptables actuellement et dans le futur.

D.6 Questionnaire

Le questionnaire a un double objectif, décrire la population pour connaître les futurs utilisateurs, mais aussi mieux cerner l'utilisation réelle de Bramanet et des transcripteurs braille en général, grâce à des questions sur le type de matériel et de logiciel employé.

Questionnaire :

Bonjour,

Nous sommes étudiants en ergonomie à l'Université de Paris 8. Nous réalisons un projet de conception d'un transcripteur universel. Ce logiciel est similaire à Bramanet, DBT ou winbraille. Il sera gratuit et téléchargeable sur Internet. Le but est d'aboutir à un logiciel pouvant transcrire différentes écritures (littéraires, mathématiques, physiques, musicales…) en Braille intégral ou abrégé, et fonctionnant sous Windows, linux ou mac.

Pour nous aider dans ce projet, nous comptons sur votre participation, afin de connaître les besoins des futurs utilisateurs. Notre but étant de déboucher sur un logiciel utile, utilisable et adapté.

L'objectif de ce questionnaire est de recueillir vos impressions sur le ou les transcripteurs que vous utilisez puis de vous demander de critiquer le logiciel Bramanet, que vous pouvez télécharger à l'adresse:

http://handy.univ-lyon1.fr/projets/bramanet/

Dans un premier temps, pouvez-vous naviguer dans ce logiciel, le tester avec vos propres documents. Il serait intéressant d'utiliser l'ensemble des fonctions, pour vous construire une représentation globale du logiciel.

Dans un second temps, pouvez-vous répondre aux questions qui suivent.

Nous vous remercions par avance de votre collaboration.

1- **Age ?** :

Permet de vérifier si ce critère rentre en ligne de compte dans l'utilisation de matériel technique.

2- **Etes vous voyant ou non voyant ?** :

Cela permet de voir si cela crée une différence au niveau des logiciels utilisés, mais aussi l'appréciation de ceux-ci.

3- **Si vous êtes non-voyant l'êtes-vous de naissance ?** :

Permet de contrôler si cela influence l'utilisation à travers la préexistence de représentations visuelles.

4- **Si non, avez vous déjà vu une interface informatique moderne et vous en souvenez vous en ? :**

Permet de mesurer de contrôler l'effet des représentations mentales d'une interface informatique visuelle.

5- **Etes vous brailliste ? :**

Au niveau de la transcription, l'utilisation du logiciel sera plus ou moins poussée selon ce critère et le contrôle du résultat pourra être fait ou non.

6- **Etes vous transcripteur ? :**

Cela permet de cerner les exigences selon le type d'utilisation.

7- **A quelle fréquence faites vous des transcriptions et quels types de transcription faites vous (intégral, abrégé, mathématiques, musicale, langues étrangères)? :**

Grâce à cette question, on peut contrôler le niveau d'expertise et si les utilisateurs ont plutôt tendance à transcrire un type de brail, ou plusieurs, et donc à avoir une utilisation réduite ou complète du logiciel. La réponse à cette question est d'autant plus intéressante si on la croise avec les autres caractéristiques.

8- **Quel est votre niveau d'expertise en informatique (débutant, expert), vous sentez vous à l'aise avec l'outil informatique ? :**

En effet certaine personnes que cela soit du à l'age ou à autre chose, on une utilisation récente ou difficile de cette technologie.

9- **Type de matériels utilisés (synthèse vocale, plage braille, embosseuse…) :**

Des outils sont t-ils préférés à d'autre, peut être intéressant pour l'agencement de l'interface.

10- **Logiciels de transcription utilisés habituellement ? :**

11- **Quels sont leurs avantages et leurs inconvénients ? :**

12- **Avez-vous utilisé auparavant Bramanet ? :**

13- **Si oui, depuis combien de temps l'utilisez-vous ? :**

Permet un control de l'expertise.

14- **Si non, l'avez-vous testé et cela pendant combien de temps ?** :

15- **Pouvez-vous nous décrire son interface de manière générale et approfondie ? :**

Permet de s'interroger sur le type de représentation selon les caractéristiques de chaque individu.

16- **Quels sont ses avantages et ses inconvénients :**

Cette question est d'ordre qualitative, et permet de contrôler la satisfaction pour la comparer à l'éventuelle utilisation de la question qui suit.

17- **Pouvez-vous nous proposer des améliorations, utiliseriez vous ce logiciel, pourquoi ? :**

Permet un contrôle subjectif de l'utilité.

18- **Pensez-vous que tous les menus soient utiles :**

Idem

19- **Pensez-vous qu'il en *manque* :**

Idem

20-**Si vous avez des suggestions supplémentaires :**

21**- Si vous souhaitez être informés sur notre projet, vous pouvez nous laisser votre adresse mail. Il ne sera utilisé que pour vous prévenir lorsque le logiciel sera mis à la disposition de la communauté et en cas de nouvelles versions.**

Annexe E : Expressions mathématiques d'OpenOffice gérées par NAT

Opérations

$+a;\ \ a;\pm a;\mp a;\neg a;a+b;a\cdot b;a\times b;a*b;a\wedge b;a\ \ b;\frac{a}{b};a\div b;a/b;a\vee b;a\circ b$

Relations

$a=b;a\neq b;a\approx b;a|b;a\nmid b;a<b;a>b;a\simeq b;a\|b;a\perp b$

$a\leqslant b;a\geqslant b;a\sim b;a\equiv b;a\leq b;a\geq b;a\propto b;a\rightarrow b;a\Leftarrow b;a\Leftrightarrow b;a\Rightarrow b$

Ensembles

$a\in b;a\notin b;a\ni b;a\cap b;a\cup b;a\setminus b;a/b;a\subset b;a\subseteq b;a\supset b;a\supseteq b;a\not\subset b;a\not\subseteq b;a\not\supset b;a\not\supseteq b$

$\emptyset;\aleph;\mathbb{N};\mathbb{Z};\mathbb{Q};\mathbb{R};\mathbb{C};$

Parenthèses et crochets

$(a);[a];[\![a]\!];|a|;\|a\|;\{a\};\langle a\rangle;\langle a|b\rangle$

Fonctions

$\lim a+b;\sum a;\prod a;\coprod a;\int a;\iint a;\iiint a;\oint a;\oiint a;\oiiint a;\int_a^b c;\int_{a=1} b;\int^a b$

$e^a;\ln(a);\exp(a);\log(a);a^b;\sqrt{a};\sqrt[a]{b};|a|;a!;$

$\sin(a);\cos(a);\tan(a);\cot(a);\arcsin(a);\arccos(a);\arctan(a);\operatorname{arccot}(a);$

$\sinh(a);\cosh(a);\tanh(a);\coth(a);\operatorname{arsinh}(a);\operatorname{arcosh}(a);\operatorname{artanh}(a);\operatorname{arcoth}(a);$

Signes divers

$\infty;\partial;\nabla;\exists;\forall;\hbar;\Re;\Im;\wp;\leftarrow;\rightarrow;\uparrow;\downarrow;\ldots;\cdots;\vdots;\ddots;$

$\wedge;\not\propto;\in;\equiv;\infty;\notin;\neq;\vee;‰;\gg;\ll;\rightarrow$

Attributs

$\acute{a}\,;\grave{a}\,;\check{a}\,;\breve{a}\,;\mathring{a}\,;\vec{a}\,;\overrightarrow{ab}\,;\tilde{a}\,;\widetilde{ab}\,;\hat{a}\,;\widehat{ab}\,;\bar{a}\,;\overline{ab}\,;\dot{a}\,;\ddot{a}\,;\underline{a}\,;\not{a}\,;\;\;;\boldsymbol{a}\,;a\,;$

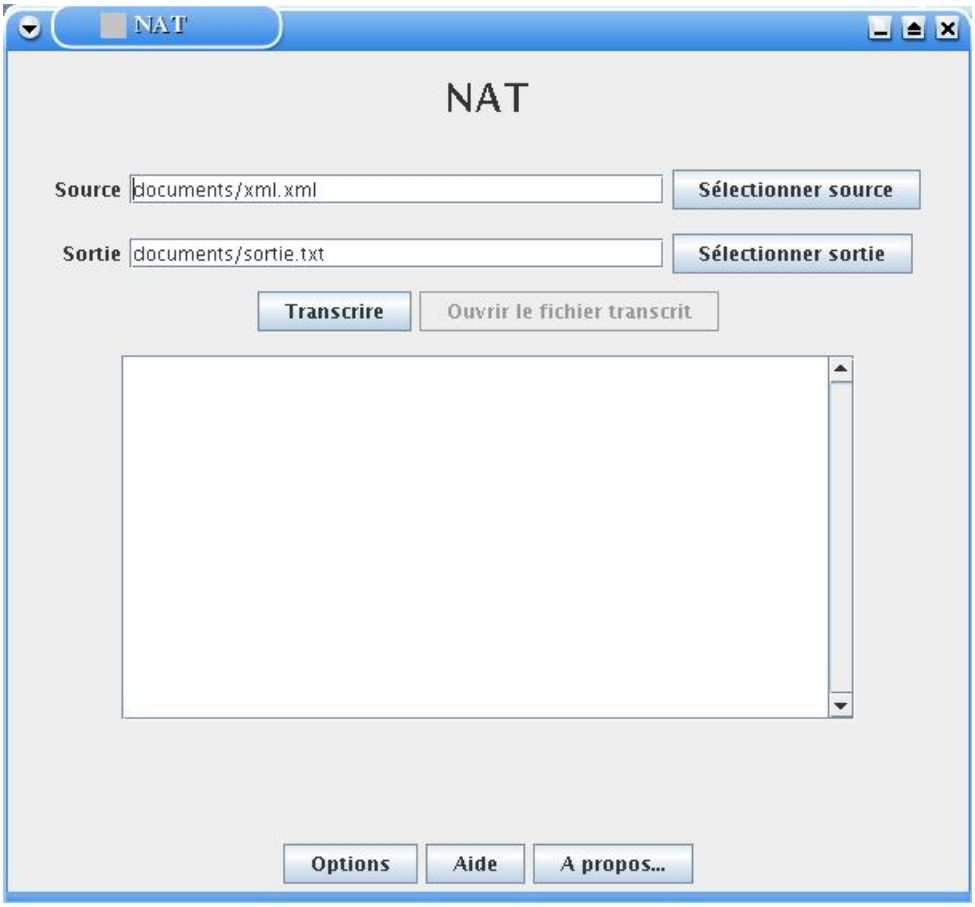
Alphabets et lettres particulières

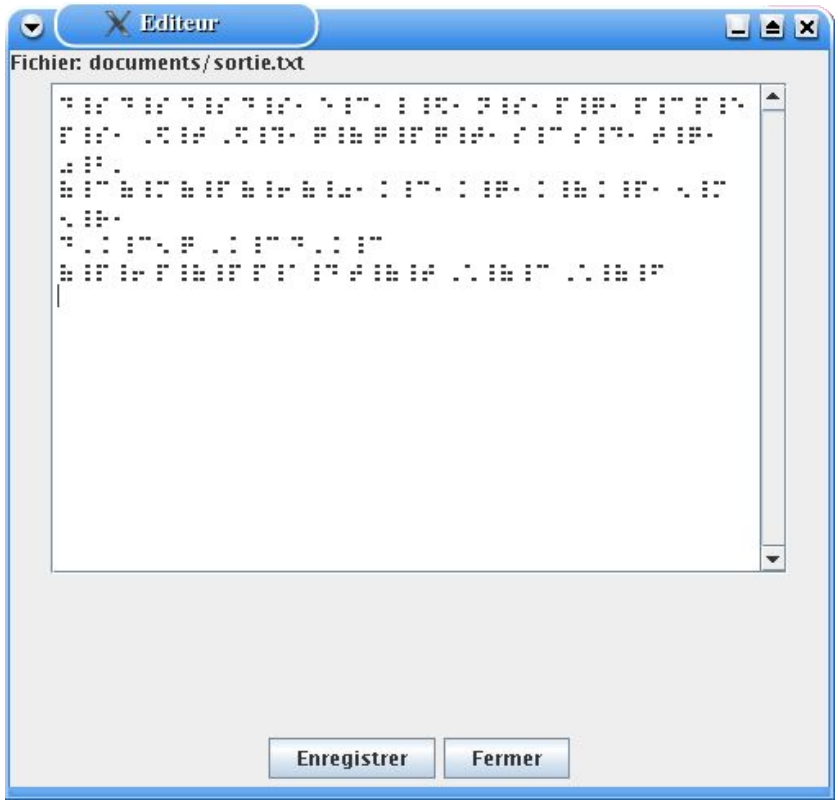
$\alpha\,;\beta\,;\gamma\,;\delta\,;\epsilon\,;\zeta\,;\eta\,;\theta\,;\iota\,;\kappa\,;\lambda\,;\mu\,;\nu\,;\xi\,;o\,;\pi\,;\rho\,;\sigma\,;\tau\,;\upsilon\,;\phi\,;\chi\,;\psi\,;\omega\,;$

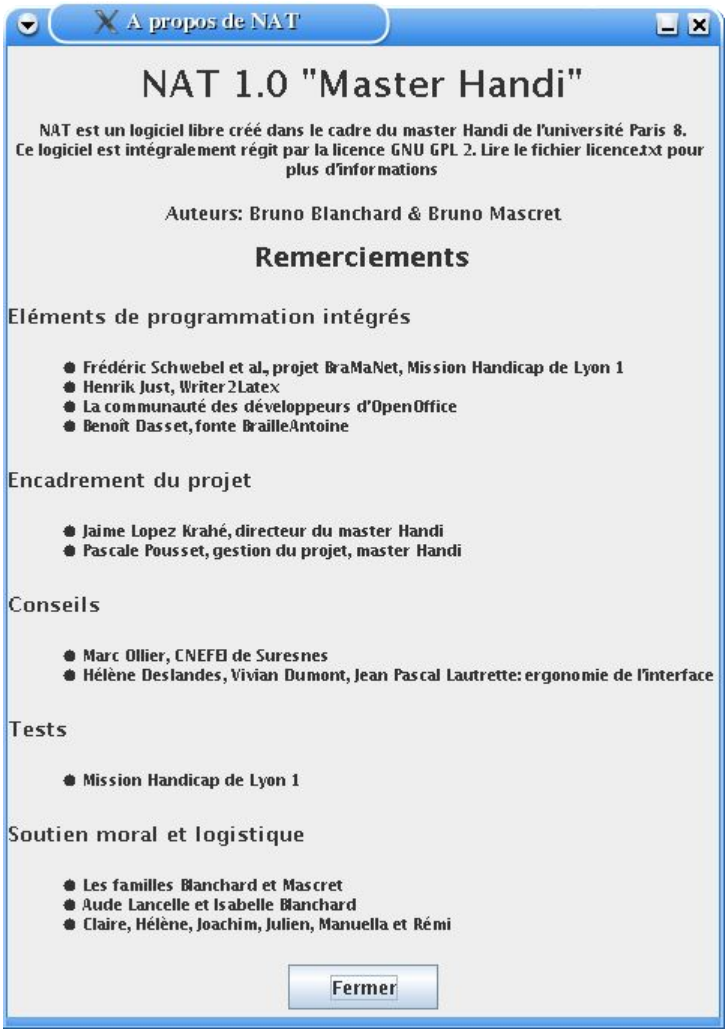
$A;B;\Gamma\,;\Delta\,;E\,;Z\,;H\,;\Theta;I\,;K\,;\Lambda\,;M\,;N\,;\Xi\,;O\,;\Pi\,;P\,;\Sigma\,;T\,;\Upsilon\,;\Phi\,;X\,;\Psi\,;\Omega\,;$

$\varepsilon\,;\varphi\,;\varpi\,;\varrho\,;\varsigma\,;\vartheta\,;$

Annexe F : Copies d'écran




Editeur
Fichier: documents/sortie.txt
Enregistrer
Fermer


A propos de NAT
NAT 1.0 "Master Handi"
NAT est un logiciel libre créé dans le cadre du master Handi de l'université Paris 8.
Ce logiciel est intégralement régit par la licence GNU GPL 2. Lire le fichier licence.txt pour plus d'informations
Auteurs: Bruno Blanchard & Bruno Mascret
Remerciements
Eléments de programmation intégrés
Frédéric Schwebel et al., projet BraMaNet, Mission Handicap de Lyon 1
Henrik Just, Writer2Latex
La communauté des développeurs d'OpenOffice
Benoît Dasset, fonte BrailleAntoine
Encadrement du projet
Jaime Lopez Krahé, directeur du master Handi
Pascale Pousset, gestion du projet, master Handi
Conseils
Marc Ollier, CNEFEI de Suresnes
Hélène Deslandes, Vivian Dumont, Jean Pascal Lautrette: ergonomie de l'interface
Tests
Mission Handicap de Lyon 1
Soutien moral et logistique
Les familles Blanchard et Mascret
Aude Lancelle et Isabelle Blanchard
Claire, Hélène, Joachim, Julien, Manuella et Rémi
Fermer