SVG & XSLT : une portée notable

Karim Barkati, Emmanuel Saint-James

Journées d'Informatique Musicale 2003

SVG & XSLT : une portée notable

Karim Barkati
Emmanuel Saint-James
Université Pierre et Marie Curie
4, place Jussieu, 75252 Paris Cedex 5, France
barkazik@club-internet.fr, Emmanuel.Saint-James@lip6.fr

Mots-clés : Partitions musicales, XML, SVG.

Résumé

La représentation de partitions musicales sur un écran, particulièrement à travers un navigateur, n'a aujourd'hui pas de solution totalement satisfaisante : l'esthétique, l'encombrement, l'indexation ou l'interactivité sont quatre critères dont l'un au moins est sacrifié au profit des autres.

Dans cette étude nous montrons qu'un jeu de feuilles de style XSL et une méta-description par DTD permettent de générer un document SVG satisfaisant les quatre critères retenus.

1 Introduction

Un nombre croissant de sites web donne accès à des partitions musicales, mais leur visualisation rapide, lisible et indépendante du navigateur est contrecarrée par une triple absence :

- une méta-description universelle de ce qu'est une partition ;
- des feuilles de styles paramétrant la mise en page ;
- un format de sortie vectoriel.

Le but de ce travail est d'y remédier, en utilisant des normes internationales.

A l'heure actuelle, les méta-descriptions utilisables sur le Web sont les DTD (*Document Type Definition*) ou les XMLSCHEMA. Nous expliquons dans la première section pourquoi les premières gardent leur avantages.

La description d'une partition par un langage de balisage comme XML pose plusieurs problèmes liés à leur structure pas toujours parenthétique (en particulier les coulés entre portées). La deuxième section de notre étude montre que la DTD nommé MUSICXML proposée par *Recordare* (www.musicxml.org) est à ce jour la plus complète de celles disponibles.

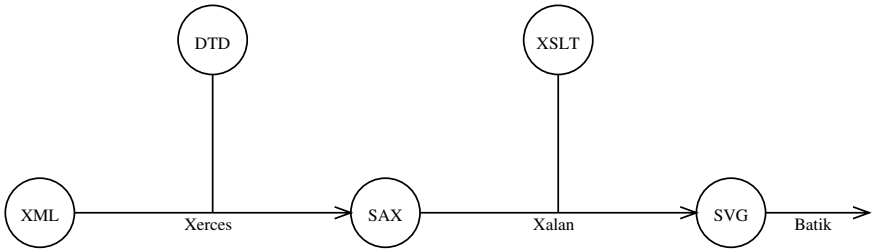
A partir d'une DTD et d'un document la respectant, on construit une représentation abstraite du document pour appliquer des opérations indépendantes de son format d'origine. Les représentations les plus connues sont DOM et SAX. La troisième section motive les choix de SAX dans un contexte de visualisation sans interaction, et de la bibliothèque XERCES pour l'obtenir.

Le jeu de feuilles de style constitue notre contribution la plus originale dans la résolution du problème visé. Après une discussion des limites de CSS, la

quatrième section montre comment énoncer en XSLT, et indépendamment de tout langage de programmation, les règles de typographie de la gravure musicale. Nous les mettons en o'euvre à l'aide de la bibliothèque XALAN appelée par un minuscule programme JAVA qui applique les règles indiquées.

Le résultat obtenu est en SVG, le language de dessin vectoriel compatible avec XML et recommandé par le W3C (www.w3.org/Graphics/SVG). Déjà opérationnel sous forme de *plug-in* dans les navigateurs et systèmes d'exploitation commerciaux, il l'est également dans les systèmes gratuits grâce à la bibliothèque BATIK d'APACHE.

Ce travail nécessite donc 5 formats, manipulés à l'aide de 3 bibliothèques :



2 Technique de méta-description

Un document XML valide obéit à une structure prédéfinie par la DTD (Document Type Definition ou Définition de type de document). Les DTD sont réutilisables, souvent libres de droits, et elles rendent la validation automatique. Par ailleurs, une feuille de style se définit toujours par référence à la structure des documents à formater : si les documents sont valides, la même feuille de style s'appliquera à tous.

Les DTD, le langage de schéma actuel pour spécifier le contenu et la structure de documents XML, souffrent de quelques déficiences. Tout d'abord, elles ne sont pas écrites en XML, ce qui signifie que les technologies existantes pour manipuler des documents XML ne peuvent être utilisées pour analyser les DTD elles-mêmes. Ensuite, elles n'ont pas d'espaces de nom, c'est-à-dire que deux éléments homonymes provenant de DTD différentes peuvent entrer en conflit, ce qui rend impossible l'import de définitions externes supplémentaires afin de réutiliser du code existant. Enfin, et surtout, les DTD n'offrent qu'un typage très limité des données. Le W3C a récemment proposé un nouveau langage de définition de schéma : XMLSCHEMA. Conçu pour palier ces déficiences, XMLSCHEMA offre en outre un intéressant mécanisme d'héritage d'attributs. Toutefois, la complexité des XML Schemas ralentit sensiblement leur adoption, et les outils pour les DTD restent plus nombreux et plus matures. En outre, il est toujours possible d'appliquer des feuilles de style sur un fichier XML dont la structure est décrite par un XML Schema.

L'extrait suivant (expurgé de ses commentaires) de la DTD *attributes.dtd*

parmi les onze qui composent l'ensemble MusicXML, montre qu'une *clef* musicale peut être définie comme un élément fils d'un élément *attributes*, et contenir une liste d'attributs *sign*, *line*, *clef-octave-change*. Le signe est obligatoire, et l'on peut spécifier une ligne d'accroche ainsi qu'une octaviation de la clef.

```
<!ELEMENT attributes (divisions?, key?, time?, staves?, instruments?, clef*,
staff-details*, transpose?, directive*)>
<!ELEMENT clef (sign, line?, clef-octave-change?)>
<!ATTLIST clef number CDATA #IMPLIED >
<!ELEMENT sign (#PCDATA)>
<!ELEMENT line (#PCDATA)>
<!ELEMENT clef-octave-change (#PCDATA)>
```

3 Méta-description d'une partition

Plusieurs formats de description de partitions musicales ont été proposés ; citons : NIFF (Notation Interchange File Format) ; CMN (Common Music Notation) ; Enigma (le format du logiciel Finale) ; MusiTeX (une format pour LaTeX) ; abc (un format en ASCII) ; HumDrum (un format ASCII pour des outils de recherche d'informations) ; MuseData (le format d'un projet de base de données musicales).

Un accès uniforme aux fonds utilisant ces formats est donc une nécessité. XML est méta-langage conçu pour supporter la définition de langages spécifiques à des communautés différentes. Extensible et modulaire, déclaratif et structuré, lisible facilement à la fois par un humain et un ordinateur (par sa grammaire formelle fondée sur Unicode), disposant de nombreux outils gratuits, ouvert et indépendant de toute plate-forme, XML est un bon candidat pour la représentation musicale. Ainsi, plusieurs formats musicaux reposant sur XML ont vu le jour (à noter qu'un seul, le dernier, utilise un XMLSchema) : SMDL (Standard Music Description Language) ; MNML (Musical Notation Markup Language) ; MHTML (Music HyperText Markup Langage) ; MML (Music Markup Language) ; XScore (eXtensible Score Language) ; NIFFML ; ChordML ; SMIL (Synchronized Multimedia Integration Language) ; MusicXML, de Recordare ; MusiXML, de Gerd Castan.

Notre choix s'est porté sur MusicXML pour plusieurs raisons. C'est un format d'échange pour les applications de notation, d'analyse, de fouille, et de performance. Il est compatible avec le format de l'éditeur de partition le plus répendu, Finale, et repose sur les formats MuseData et Humdrum. Comme XML est un format hiérarchique, il n'est pas possible d'encoder la nature bidimensionnelle des partitions par une notation de la présentation en deux dimensions. Pour y parvenir, Humdrum représente explicitement cette propriété par deux formats de haut niveau : partwise.dtd, qui contient les mesures à l'intérieur de chaque voix (*parts* en anglais) et timewise.dtd, qui contient les voix à l'intérieur de chaque mesure

Deux feuilles de style XSLT (parttime.xsl et timepart.xsl) sont fournies pour passer d'un format à l'autre. Les DTD de partitions en timewise ou en partwise ne représentent qu'un mouvement d'une pièce. Plusieurs mouvements, ou toute

autre collection musicale, peuvent être représentés à l'aide de la DTD opus.dtd. Le document opus contient des liens XLink vers des partitions individuelles, et évoluera vers l'inclusion de références plus détaillées et d'informations musicologiques. Il existe aussi une DTD midixml.dtd pour la représentation des fichiers MIDI en XML. MusicXML est donc un format très complet, ce qui constitue une autre des raisons de notre choix. Une page de musique imprimée pour chant et piano se code en environ à 100 ko en MusicXML.

Voici un exemple minimaliste d'une partition ne contenant qu'une voix (*part*) et qu'une mesure (*measure*), avec une clef de sol :

```
<?xml version"1.0" encoding"UTF-8" standalone"no"?>
<!DOCTYPE score-partwise PUBLIC
 "-//Recordare//DTD MusicXML Partwise//EN"
 "http ://www.musicxml.org/dtds/partwise.dtd">
<score-partwise>
    <movement-title>Une Clef</movement-title>
    <part-list>
      <score-part id"P1">
        <part-name>Partie 1</part-name>
      </score-part>
    </part-list>
    <part id"P1">
      <measure number"1">
        <attributes>
          <clef>
            <sign>G</sign>
            <line>2</line>
          </ clef>
        </attributes>
      </measure>
    </part>
</score-partwise>
```

4 La représentation abstraite

Les documents XML structurent leurs données en une arborescence complexe grâce à un système de balisage. Selon le type d'utilisation souhaitée, deux façons d'analyser un document XML sont disponibles : DOM et SAX.

Le DOM (Document Object Model), est une norme du W3C qui spécifie une interface permettant d'écrire des fonctions de création, de modification, et de consultation des éléments constitutifs d'un document XML ou HTML. Il est défini indépendamment d'une plateforme ou d'un langage. Le modèle de représentation de document utilisé par le DOM est le *bosquet*, un graphe composé d'arbres et d'arcs reliant certains noeuds d'un arbre à un autre. Le DOM se destine à la création et à la manipulation de données représentant un arbre, en proposant des méthodes qui permettent de manipuler l'arbre ou de le parcourir efficacement.

SAX signifie Simple API for XML. Une *Interface pour la Programmation d'Applications* est un ensemble de fonctions et d'objets réutilisables. SAX est un standard de fait, un produit de la coopération spontanée et volontaire des

développeurs sur Internet (cf http ://www.xml.org). Ceux-ci constataient en 1997 que les différents phraseurs (*parser* en anglais) expérimentaux alors disponibles offraient tous des API différentes, et que cette diversité interdisait de changer facilement de phraseur lorsqu'un nouveau était proposé. Regroupés sur une liste de discussion [XL-dev], ces développeurs décidèrent alors de définir une API commune, décrite en Java, et une implémentation de référence, mise dans le domaine public. SAX est maintenant reconnu par la quasi-totalité des phraseurs du marché fondés sur le modèle évènementiel. En effet, il fonctionne sur un système de flux évenementiel, c'est-à-dire que lors de la lecture d'un flux XML, des évènements sont émis vers l'application. SAX est plus performant que DOM, mais moins complet, et se destine donc surtout à la lecture de documents XML, ce qui suffit à notre projet dans sa phase actuelle.

De nombreux analyseurs de documents XML existent et parfois coexistent au sein d'un même projet ; citons : Apache's XML Project ; Sun's Java Project X Package ; IBM's XML4J, Oracle's XML Package DataChannel's XJParser Package ; Microstar's AElfred Parser ; James Clark's XP Parser. Le site http ://xml.apache.org héberge l'Apache XML Project, un ensemble très complet d'outils XML gratuits pour les développeurs, dont Xerces, un phraseur XML, disponible en Java et C++. Le phraseur Java Xerces, comportant des centaines de classes, respecte la recommandation XML 1.0 et contient d'autres fonctionnalités avancées, comme XML Schema, DOM niveau 2, et SAX version 2, en plus de supporter les API standardisées DOM niveau 1 et SAX version 1. Xerces continue d'évoluer, notammant vers le support complet de XML Schema. Par souci de cohérence avec Batik et Xalan du même Apache, nous avons choisi d'utiliser la bibliothèque Xerces pour traiter les documents XML, bien que XML4J d'IBM semble aussi excellent. Une première comparaisaon, ainsi qu'un test positif de chargement et d'analyse de fichier MusicXML confirment la validité de notre choix.

5 Les feuilles de style

Incluses dans les feuilles de style, les propriétés graphiques ou typographiques du document final sont séparées du document source. Il existe plusieurs langages de spécification de feuilles de style, dont CSS1, CSS2, XSL. Ils permettent tous d'indiquer les modalités de réalisation physique de chacun des éléments qui composent un document.

5.1 Limites de CSS

CSS (Cascading Style Sheets ou Feuilles de style en cascade) est un mécanisme puissant et relativement complexe pour gérer des styles (par exemple les polices, les couleurs, l'espacement) des documents Web. Les feuilles de style CSS sont donc utilisées pour la mise en page sur le Web comme méthodes efficaces de définitions et d'organisation des styles pour l'affichage du texte et des graphismes. Les CSS ont été initialement conçues pour le langage HTML. La

première version CSS1 a été publiée en 1996 par le W3C, et la seconde, CSS2, en 1998. Celle-ci introduit notamment la notion de type de média qui donne la possibilité de spécifier la mise en pages d'un document en vue de son affichage sur écran, de son impression sur papier, ou d'indiquer les conditions de sa réalisation par synthèse vocale. SVG utilise CSS2 pour inclure des feuilles de style en ligne, externes, ou integrées. Tout ce qui concerne les attributs de mise en page (comme les espacements) et les attributs graphiques (comme le remplissage) peut être défini en tant que style. Techniquement, les CSS sont définies en deux parties : le sélecteur et le style. Dans `.rectangle {fill :red ;}` par exemple, `rectangle` est le sélecteur et `fill :red` est le style.

Les styles permettent d'enchaîner en cascade des changements à travers un document ou un site en éditant la définition. Toutefois, ces directives locales sont insuffisantes pour mettre au point une justification globale d'une partition.

5.2 Les composantes de XSL

XSL (eXtensible Stylesheet Language ou Langage extensible de feuille de style) est un langage pour écrire des feuilles de style applicables sur des documents XML. Il est divisé en trois parties :

- XSLT (XSL Transformations) : un langage pour la transformation de documents XML
- XPath (XML Path Language) : un langage d'expression utilisé par XSLT pour accéder ou référencer des parties d'un document XML. XPath est aussi utilisé par XLink, la spécification de liens XML.
- XSLFO (XSL Formatting Objects) : un vocabulaire XML qui spécifie des sémantiques de formatage.

Une feuille de style XSL spécifie la présentation d'une classe de documents XML, en décrivant comment une instance de la classe est transformée en un document XML qui utilise le vocabulaire de formatage. A la différence de CSS, XSL utilise lui-même une notation XML.

XSLT (http ://www.w3.org/TR/xslt) est un langage utilisé pour transformer des documents XML en d'autres documents XML, ce qui est le cas de nos documents MusicXML, notre format d'entrée, et des fichiers SVG, notre format de sortie. XSLT est conçu pour être utilisé comme une partie de XSL, le langage des feuilles de style de XML. XSL spécifie les règles de présentation d'un document XML en utilisant XSLT pour décrire comment le document peut être transformé en un autre document qui utilise le vocabulaire de formatage. XSLT est aussi conçu pour être utilisé indépendamment de XSL. Ainsi, nous utilisons XSLT pour transformer des documents MusicXML en documents SVG, tous deux définis en XML, et pour stocker les règles de transformation, dont les règles de mise en page.

XPath (http ://www.w3.org/TR/xpath) est un langage d'expression pour le référencement des parties d'un document XML. Par exemple, il permet l'accès aisé à n'importe quel no'eud d'un document, l'entrecroisement, et la consultation d'autres no'euds que celui courant pour paramétrer l'application d'une règle.

XLink est la norme de spécification des liens XML, à l'intérieur d'un document comme à l'extérieur, en utilisant XPath. Nous l'utilisons entre autres pour référencer les polices textuelles, ainsi que les glyphes musicaux. Pour pouvoir créer des liens dans une feuille de style XSLT, il faut préalablement déclarer XLink de la façon suivante au début du fichier :

```
<?xml version"1.0"?>
<xsl :stylesheet version"1.0
      xmlns :xsl"http ://www.w3.org/1999/XSL/Transform"
      xmlns :xlink"http ://www.w3.org/1999/xlink">
```

5.3 L'énoncé des règles typographiques

XSLT permet de spécifier des règles de transformation indépendamment du langage de programmation utilisé pour le moteur de transformation un peu comme Yacc. D'une certaine manière, cela revient à faire de la programmation dirigée par les données, et se révèle approprié à la complexité des structures de données musicales.

Nous avons choisi Java pour écrire notre moteur d'application des règles, avec la bibliothèque Xalan d'Apache, qui propose précisément des processeurs de feuille de style XSLT, disponibles en Java et C++ Xalan implémente complètement les recommandations XSLT et XPath du W3C. Le processeur de feuille de style est robuste et propose de nombreuses fonctionnalités. Le processeur XPath est utilisable de façon indépendante. Xalan utilise le Bean Scripting Framework (BSF) pour implémenter les extensions de Java ou des autres langages de script.

Xalan propose entre autres un processeur XSLT en Java, formées de quelques centaines de classes. Ce processeur parcours un document XML, et lorsqu'il trouve un motif correspondant à l'un des motifs de la feuille de style, il applique les règles associées. Ainsi, le processeur Xalan construit un arbre-résultat à partir d'un arbre-source. Ce couple processeur / feuille de style permet de manipuler la structure logique des documents XML, ce dont nous avons besoin pour développer notre moteur de conversion MusicXML vers SVG. En effet, il s'agit de constuire un arbre-résultat SVG contenant les éléments graphiques à partir des éléments syntaxiques contenus dans un arbre-source MusicXML.

```
import org.apache.xalan.xslt.XSLTProcessorFactory ;
import org.apache.xalan.xslt.XSLTInputSource ;
import org.apache.xalan.xslt.XSLTResultTarget ;
import org.apache.xalan.xslt.XSLTProcessor ;
[...]
  XSLTProcessor processor XSLTProcessorFactory.getProcessor() ;
  processor.process(new XSLTInputSource(inName),
                    new XSLTInputSource(xslName),
                    new XSLTResultTarget(outName)) ;
```

Ce processeur XSLT nous est aussi utile pour changer de représentation MusicXML : partwise ou timewise. De fait, en amont, MusicXML utilise déjà deux feuilles de style parttime.xsl et timepart.xsl, respectivement pour passer d'une représentation par voix (verticalement sur la partition d'orchestre) à une représentation par mesures (horizontalement), et réciproquement.

5.4 Un exemple de feuille de style

Voici un extrait d'une feuille de style XSLT. Il s'agit d'une règle typographique qui indique le code SVG à générer lorsqu'une clef de sol est rencontrée. Un patron (*template*) vérifie que l'élément est une *clef*, et qu'il possède un attribut *sign* de valeur *G*. Ce patron contient ensuite les règles de création d'un élément XML *use* doté de deux attributs et d'une valeur textuelle, ce qui correspondra à du code SVG une fois cette règle appliquée par un moteur de transformation sur un document XML.

```
<xsl :template match"clef">
    <xsl :if test"sign'G'">
      <xsl :element name"use">
        <xsl :attribute name"xlink :href"
             namespace"http ://www.w3.org/1999/xlink">
          <xsl :text>MusicGlyphs.svg#GKEY</xsl :text>
        </xsl :attribute>
      </xsl :element>
    </xsl :if>
  </xsl :template>
```

Toutes les règles de transformations étant contenues dans la feuille de style, celle-ci peut rapidement avoir une taille importante (une dizaine de pages), alors que le moteur d'application des règles, reposant sur les librairies Xalan et Xerces déjà écrites, garde sa taille étonnamment compacte (deux pages de code Java).

6 Le document graphique

SVG (Scalable Vector Graphics ou Graphismes vectoriels redimensionnables) est un langage de description des graphismes bi-dimensionnels en XML. Héritier de PostScript, il trace des contours pour dessiner aussi bien des images que du texte. Les objets graphiques peuvent être groupés, stylés, transformés et recomposés dans les objets rendus précédemment. Du texte peut être inséré dans n'importe quel espace de noms XML de l'application, ce qui renforce les possibilités de recherche et d'accès des graphismes SVG. L'ensemble de fonctionnalités inclut les transformations emboîtées, les pochoirs, les masques alpha, les effets de filtre, les patrons, et une certaine extensibilité. Les dessins SVG peuvent être dynamiques et interactifs. Le DOM de SVG, qui inclut le DOM XML complet, permet de faire de l'animation graphique vectorielle via des scripts. Un ensemble de fonctions de prise en charge d'évenements tels que *onmouseover* et *onclick* peut être assigné à n'importe quel objet graphique SVG. Grâce à sa compatibilité avec d'autres standards du Web (http ://www.w3.org/Graphics/SVG), des fonctionnalités comme l'application d'un script peuvent être faites simultanément sur des éléments SVG et d'autres éléments XML provenant d'espaces de noms différents, au sein de la même page Web. Nous utilisons SVG comme format de sortie pour la visualisation des partitions sur le Web. Deux applications intéressantes de ce format sont envisageables :

- le remplacement des polices musicales (inconnues des navigateurs) alors qu'un programme SVG peut embarquer tous les symboles nécessaires ;

– l'intégration de la notation musicale sur le Web, avec à la fois des fichiers de petite taille, un affichage de haute qualité, et l'impression de haute qualité pour ces mêmes données.

Voilà un court exemple de code SVG généré par notre moteur d'application des règles de transformation XSLT sur un document MusicXML constitué d'une clef de sol. La première ligne indique qu'il s'agit d'un document XML, puisque SVG est méta-décrit par XML, puis la deuxième spécifie l'emplacement de la DTD de SVG. Ce document contient ensuite une balise *text* et sa valeur, définit ce qu'est une portée avec la balise *defs*, puis place une instance de cette portée dans un repère pixelisé commençant en haut et à gauche (comme X-window, alors que PostScript commence en bas). Il place enfin une instance d'un élément SVG nommé *GKEY* (une clef de sol), un glyphe référencé dans le fichier MusicGlyph.svg par XPath.

```
<?xml version"1.0" encoding"UTF-8"?>
<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 20001102//EN"
"http ://www.w3.org/TR/2000/CR-SVG-20001102/DTD/svg-20001102.dtd">
<svg xmlns :xlink"http ://www.w3.org/1999/xlink" width"300" height"300">
    <text x"150" y"62"
     style"font-family :Times Roman; font-size :30;
     text-anchor :middle">Une Clef</text>
    <defs>
        <g id"Staff1" style"stroke-width :1;stroke :black">
            <line x1"20" x2"280" y1"0" y2"0"/>
            <line x1"20" x2"280" y1"14" y2"14"/>
            <line x1"20" x2"280" y1"28" y2"28"/>
            <line x1"20" x2"280" y1"42" y2"42"/>
            <line x1"20" x2"280" y1"56" y2"56"/>
        </g>
    </defs>
    <g transform"translate(0,170)">
        <use xlink :href"#Staff1"/>
    </g>
    <g transform"translate(30,154)">
        <use xlink :href" MusicGlyphs.svg#GKEY"/>
    </g>
</svg>
```

Une fois obtenu le code SVG, il faut le passer à un visualiseur approprié. Batik (http ://xml.apache.org/batik/) est un ensemble d'outils Java utilisables individuellement ou collectivement pour visualiser, générer ou manipuler du SVG. Batik est extensible : par exemple, il permet au développeur d'employer ses propres balises SVG. L'un des programmes fourni avec cet ensemble est un visualiseur SVG, qui valide les différents modules et leur inter-opérabilité, et rend SVG opérationnel sous Linux, dans un contexte de gratuité des logiciels. Sur les plateformes Macintosh et Windows, le plug-in SVGViewer d'Adobe permet de visualiser les fichiers SVG au sein des navigateurs Netscape et Explorer. A terme, SVG devrait être supporté directement par les navigateurs, tout comme XML, afin de dépasser les limites imposées par le bitmap et HTML.

7 Conclusion

Nos expérimentations ont donc montré que les outils actuellement disponibles autour de XML sont déjà suffisants pour représenter avec une qualité maximale un fonds de partitions non nécessairement orienté vers l'édition au départ. On peut donc raisonnablement penser que le passage à XML pour les partitions est appeler à se généraliser. Il reste à concevoir un outil d'édition musicale interactive et coopérative autour de SVG pour tirer tout le parti offert par ces avancées technologiques.

8 Bibliographie

Typographie Coopérative de Partitions Musicales par Navigation Interprétative de Contours, thèse de doctorat de l'Université Paris 6 spécialité informatique de Nabil Bouzaiene, juillet 2000.

The Java Programming Language Third Edition ; Ken Arnold, James Gosling, David Holmes, ed. Sun Microsystem 2000

Programmation Java, Jean-François Macary et Cédric Nicolas, Eyrolles 1996

XML and Java : Developing Web Application ; Hiroshi Maruyama, Kent Tamura, Naohiko Uramoto (IBM), ed. Addison-Wesley 1999

Unleashed XML ; Michael Morrison, ed. Sams Publishing 2000

SVG 1.0 Specification, W3C Candidate Recommandation 2000

XML Langage et applications, Alain Michard, Eyrolles 3e tirage 1999

Le programmeur XML, Simon North et Paul Hermans, CampusPress 1999

XML Schema Part 0 :Primer ; W3C 30 mars 2001

De l'écrit au numérique -constituer, normaliser et exploiter les corpus électroniques- Benoît Habert, Cécile Fabre, Fabrice Issac, InterEditions Masson 1998 par