LXir 1.0 : guide de l'utilisateur

Jean-Paul Jorda & Xavier Trochu – EDP Sciences

24 septembre 2007

Version de ce document: 0.1

Table des matières

Introduction

Cette documentation a pour but de vous aider à utiliser et à configurer LaTeXir.

LaTeXir est un convertisseur permettant de produire du XML/MathML à partir d'un document LaTeX. Il a été développé par l'éditeur scientifique EDP Sciences[1] dans le cadre d'un collaboration tripartie (Cyberthèse/EDP Sciences/AJLSM) menée par *Cyberthèse* [2]. L'objectif de cette collaboration est de permettre la publication en ligne de thèses composées avec LaTeX en utilisant la plate-forme Cyberdocs, développée par la société AJLSM[3]. Pour concevoir LaTeXir, nous avons emprunté certains concepts au convertisseur Hermes[4]. Néanmoins, leur mise en œuvre dans LaTeXir est différente pour permettre, entre autre, une configuration plus facile. Pour la conversion des formules mathématiques vers MathML, LaTeXir utilise une méthode originale basée sur les fichiers de log.

Il exite plusieurs outils de conversion de LaTeX vers XML, et le meilleur choix peut dépendre de ce que vous cherchez à obtenir. Aussi nous vous invitons à lire ci-dessous si LaTeXir est fait pour vous.

1 Fonctionnalité d' LaTeXir

1.1 LaTeXir est-il fait pour vous ?

La conversion automatique d'un document LaTeX vers XML peut être extrêmement compliquée. Plusieurs outils ont été développés pour faciliter ce travail, tous ayant leurs avantages et leurs inconvénients. Vous trouverez dans la bibliographie les liens vers les pages Web de la plupart de ces convertisseurs. Pour vous aider à déterminer si LaTeXir est fait pour vous, nous avons noté ci-dessous quelques points clés .

LaTeXir produit du « XML LaTeXir ». Ce XML est très complet, mais vous devrez probablement le transformer en un autre format. Si vous souhaitez un outil qui produise directement du DocBook ou du TEI, LaTeXir n'est, pour l'instant du moins, pas fait pour vous. Par contre, contrairement à d'autres outil, vous n'avez pas besoin d'être un expert TeX pour produire le XML qui vous convient.

LaTeXir est hautement configurable. C'est l'un des objectifs essentiels ayant présidé au développement d'LaTeXir. L'adaptation d'LaTeXir à une nouvelle classe ou un nouveau style est plus rapide et plus compréhensible qu'avec la plupart des autres outils de conversion de LaTeX vers XML.

LaTeXir utilise le moteur *latex*. Il n'inclut pas d'interpréteur LaTeX et a donc besoin d'une distribution TeX pour fonctionner. Par contre, la complexité des documents LaTeX que vous souhaitez convertir n'est pas limitée.

LaTeXir a été développé pour LaTeX 2ε. Il ne pourra pas convertir vos documents LaTeX2.09, TeX, ConText, MusiTeX,...

LaTeXir est jeune. Il peut donc être moins stable que d'autres convertisseurs et ses caractéristiques peuvent encore évoluer.

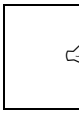
LaTeXir est « simple ». Entendons nous bien : la conversion de LaTeX vers XML est compliquée dans son principe, mais on espère que le fonctionnement et la configuration d'LaTeXir soient plus compréhensibles, plus rapides, plus simples qu'avec les autres convertisseurs.

En résumé, si vous avez un besoin ponctuel de conversion d'un document LaTeX vers une DTD bien précise, LaTeXir n'est probablement pas (encore) ce qu'il vous faut. Par

contre, si vous avez quelques compétences en XML et en LaTeX et que vous souhaitez mettre en place une chaîne de conversion de vos documents LaTeX, l'utilisation d'LXir peut être un atout intéressant.

1.2 Le « XML LXir »

Les caractéristiques essentielles du XML fournit par LXir sont résumées ci-dessous :
- il contient les informations « sémantiques » du document LaTeX d'origine ;
- il a une structure proche de celle du document d'origine, tout en étant plus hiérarchisé ;
- il n'inclut pas ou peu d'informations de mise en page ;
- la plupart des informations sémantiques sont balisées par des éléments XHTML `span`, `div` ou `a` (ex : `<div class="title">le titre</div>`) ; cela permet une visualisation avec un navigateur Web ;
- il contient du MathML pour les formules mathématiques ;
- les tableaux sont balisés en XHTML ;
- il inclut l'appel des figures d'origines, sans post-traitement.
- il n'est pour l'instant pas associé à une DTD ou un Schema ;



La documentation « de référence » concernant le XML LXir est inclus dans le document « *LXir Tags Documentation* ».

Le XML produit est susceptible d'évoluer dans les prochaines version d'LXir.

1.3 Styles et classes LaTeX pris en compte

Les classes et les style LaTeX pris en compte à ce jour selon le mécanisme brièvement décrit ci-dessous dans la section 2.4 sont présentés dans le tableau 1. L'utilisation d'autres styles est généralement possible, mais aux commandes et environnements définis ou redéfinis dans ce style ne seront simplement pas associés des balises XML, ce qui peut être gênant, ou non, selon ce que vous voulez obtenir.

Si vous avez quelques compétences en TeX/LaTeX, vous pouvez compléter ou modifier le balisage des classes et des styles qui vous intéressent (voir la section 4.1).

1.4 Fontes prises en compte

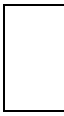
LXir convertit les glyphes utilisés dans le document en caractères UNICODE. Pour cela, LXir utilise des tables de données en XML permettant de déterminer le codage de la fonte et la position des caractères pour chaque codage. La liste des fontes prises en compte et leur encodage se trouve dans les fichiers `fonts.xml` et `fontsmath.xml`. Pour les fontes dont l'encodage est noté `DefaultLXirEncoding`, le codage n'a pas pu être déterminé : c'est OT1 qui sera utilisé.

Si vous utilisez une fonte qui n'est pas encore prise en compte par LXir, vous pouvez ajouter vos propres tables de données (voir la section 4.3).

LXir traite aussi les caractères composites (ex : `\c C`) dans la mesure ou un caractère UNICODE équivalent existe.

TAB. 1 – Liste des styles et des classes préparés pour LaTeXir.

Classe ou style	État
aa.cls	OK
article.cls	OK
book.cls	OK
report.cls	OK
array.sty	OK
natbib.sty	OK
tabularx.sty	partiel
enumerate.sty	OK
babel.sty	partiel
graphicx.sty	OK
color.sty	OK



Les styles permettant la fabrication de figures en LaTeX, utilisant picture (par exemple epic), metafont/Metapost ou PostScript (par ex. pstricks), ne sont pas gérés pas LaTeXir.

2 Une introduction au fonctionnement d'LaTeXir

2.1 Les composants d'LaTeXir.

Le convertisseur LaTeXir est constitué de plusieurs composants (voir la figure 1). *Le style lxir* sert à modifier un certain nombre de paramètres et de macros LaTeX. Avec les *fichiers de balisage des classes et des styles* (par exemple `article_lxir.sty`), ils permettent d'obtenir, après compilation avec *latex*, un fichier DVI sémantiquement balisé et un fichier de log enrichi. Le *programme lxir* va lire ces fichiers pour produire le document XML. Il utilise des *fichiers de données* (par exemple `fonts.xml`), des *fichiers de configuration* (`config.xml` et `transformations.xml`) et des *feuilles de style XSLT*.

Le fonctionnement un peu plus détaillé du processus de transformation est décrit ci-dessous.

2.2 Un DVI balisé

Le fichier DVI produit par une compilation LaTeX contient essentiellement des instructions pour le placement des glyphes et des traits sur la page. Il est donc sémantiquement très pauvre. Il est possible (cf. par exemple [7]), en modifiant des paramètres et en redéfinissant des macros, de baliser le fichier DVI. LaTeXir utilise donc notamment (tout comme tex4ht[8] et Hermes[4]), la possibilité offerte par TeX, via la commande `\special{}`, d'introduire des balises dans le fichier DVI. C'est le but principal du chargement du style lxir et des fichiers `*_lxir.sty`.

2.3 La conversion des formules mathématiques

LaTeXir utilise une méthode originale pour la conversion vers MathML. Celle-ci est basée sur l'analyse du fichier de log produit par LaTeX. Les paramètres de LaTeX sont

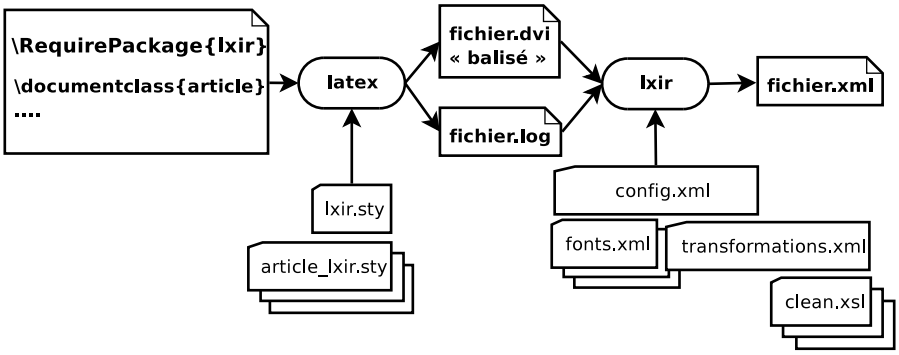


FIG. 1 – Le fonctionnement d'LXir

modifiés dans le style lxir pour rendre TEX plus bavard sur la façon dont les « boîtes » sont agencées pour composer les formules. Le programme *lxir* traduit les informations de ce fichiers de log en structures XML, et par transformations successives construit du MathML. Les formules MathML sont ensuite introduites dans le XML du document à l'aide des attributs `id` affectés à chacun des noeuds.

2.4 La redéfinition de commandes LATEX

Avec LXir, les commandes et les environnements pour lesquels on souhaite associer un balisage (et seulement celles-là) doivent être redéfinis. En LATEX, les commandes sont définies à différents niveaux (les primitives TEX, le format latex, les classes, les styles, ...). Les commande `\documentclass{}` et `\usepackage{}` permettent le chargement des classes et des styles. Le style lxir modifie ce mécanisme (voir la figure 2) de façon à introduire les redéfinitions des macros « balisées ».

2.5 Des glyphes aux caractères.

Dans le fichier DVI, chaque lettre ou symbole est décrit par un numéro représentant la position du glyphe correspondant à ce caractère (ou à ces caractères, dans le cas des ligatures) dans la fonte utilisée. Le fichier XML produit par le programme *lxir* contient de son côté des caractères UNICODE. Pour passer de l'un à l'autre, *lxir* utilise des tables de conversion permettant de déterminer le codage des fontes (`fonts.xml`) et la position de chacun des caractères pour un codage donné (`encodings.xml`).

2.6 La chaîne de traitement XML

L'analyse du DVI permet au programme *lxir* de construire des structures XML à partir des fichiers `.dvi` et `.log`. Une suite de transformations DOM et XSLT (une cinquantaine) permet ensuite d'aboutir au XML final. L'ordre et la nature de ces transformations sont décrites dans le fichier `transformations.xml`. Une part importante de ces transformations concerne la conversion vers MathML.

Si vous avez des compétences en programmation XML (DOM, XSLT,...), il vous est possible de modifier ou de compléter cette chaîne de traitement (voir la section 4.2).

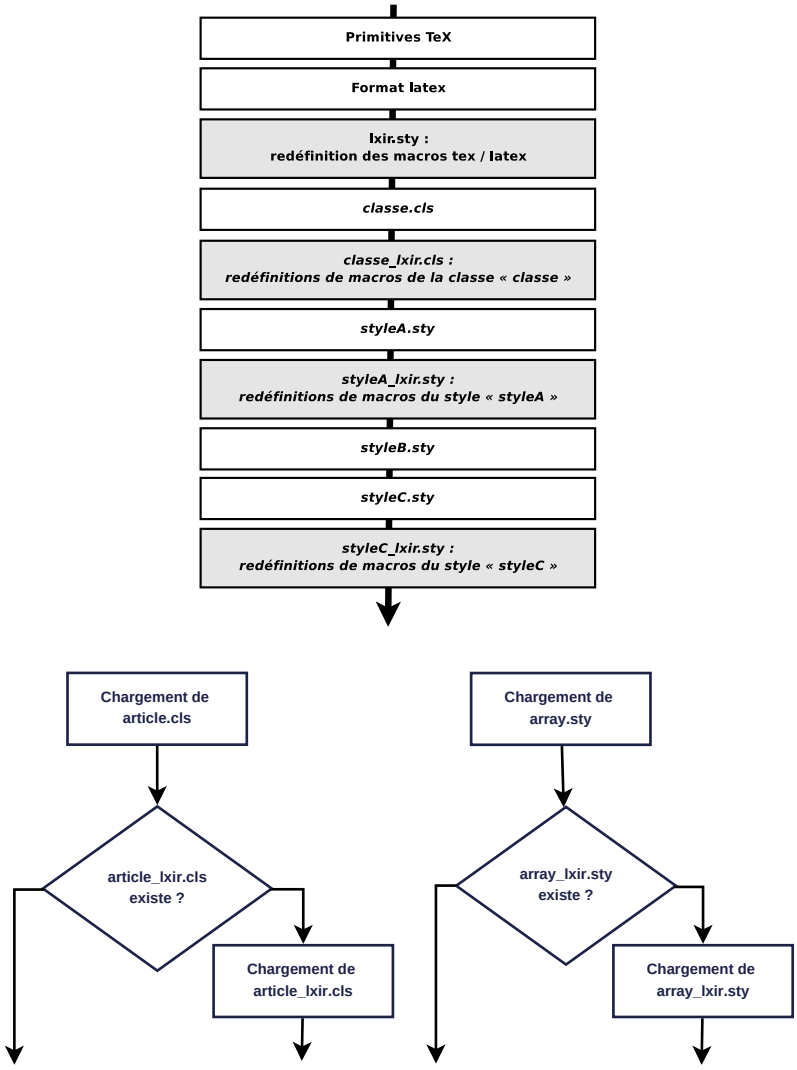


FIG. 2 – Mécanisme de redéfinition des macros définies dans les classes et les styles.

3 Utiliser LXir

3.1 Installer LXir

> ☞ **La procédure d'installation d'LXir est décrite dans le document**
> « ***LXir 1.0 : guide d'installation*** »

3.2 Comment convertir votre document ?

1. Vérifiez que votre document LaTeX compile sans erreur : c'est la condition *sine qua non* pour l'utilisation d'LXir :

```
➤ latex mydocument.tex
```

2. Ajoutez `\RequirePackage{lxir}` *avant* `\documentclass{...}`. Le paquetage lxir rendra votre document compilé impropre à la visualisation ou à l'impression. Vous pouvez donc dupliquer votre fichier principal auparavant :

```
cp mydocument.tex mydocument_lxir.tex
# et ajouter \RequirePackage{lxir}
➤ # dans mydocument_lxir.tex
# avec votre editeur favori
```

3. Compilez avec `latex` le fichier ainsi modifié :

```
➤ latex mydocument_lxir.tex
```

ou

```
➤ latex -max-print-line=1000 mydocument_lxir.tex
```

si vous utilisez une version récente de MikTeX.

4. Lancez la commande `lxir` en passant le fichier DVI en paramètre :

```
➤ lxir mydocument_lxir.dvi > mydocument.xml
```

3.3 Le script Perl *runlxir.pl*

Si le langage Perl est installé sur votre machine, vous pouvez utiliser le programme *runlxir.pl*, qui effectuera pour vous les différentes étapes du processus. Avec l'exemple ci-dessous, il suffit de lancer :

```
➤ runlxir.pl mydocument.tex
```

pour obtenir un fichier `mydocument.xml` contenant le XML LXir.

4 Configurer LXir

LXir offre aux utilisateurs la possibilité de choisir les informations à baliser (voir 4.1) et la façon dont ces informations seront structurées dans le XML (voir 4.2). Vous trouverez les fichiers de configuration dans le répertoire `TEXMF_DEST_DIR` que vous avez choisi lors de l'installation avec l'option de configuration `--with-textmf-destdir`

4.1 Baliser une classe ou un style

4.1.1 Macros pour le balisage

Le style lxirtag (chargé par le style lxir) définit un jeu de macros destinées à faciliter le balisage des informations sémantiques. Le tableau 2 fournit une description des principales commandes et de leur résultat en XML (en l'absence de modification de la chaîne de traitement XML). Le but de ces macros est d'introduire des commandes `\special{}` dont le contenu sera interprété par le programme *lxir*. Ces macros sont

TAB. 2 – Principales commandes utilisées pour le balisage.

Balisage	**Résultat en XML**
`\xBegin{mytag}...\xEnd{myTag}`	`<span class="mytag">...</span>`
`\xEmpty{mytag}`	`<span class="mytag"/>`
`\xEmptyA{mytag}{attr=val}`	`<span class="mytag" attr="val"/>`
`\xBegin{mytag}\xAttr{attr=val}` `...\xEnd{myTag}`	`<span class="mytag" attr="val">` `...</span>`

utilisables directement dans votre document, mais leur vocation première est de servir au balisage des classes et des styles LaTeX.

4.1.2 Baliser une classe ou un style

Dans la plupart des cas, baliser une classe ou un style revient à créer un fichier contenant les macros balisées. Par exemple si vous voulez baliser une classe maclasse (définie dans le fichier `maclasse.cls`), vous pouvez créer un fichier `maclasse_lxir.cls` en introduisant les commandes balisées. Ce fichier sera chargé automatiquement par *latex* lors de la compilation avec le style lxir.

En fonction de ce que voulez faire et de la complexité de votre classe ou style, le balisage sera trivial ou... très compliqué ! Schématiquement, les auteurs d'LaTeX*ir* ont jusqu'à présent utilisé trois techniques :

- Technique d'*encadrement* : la définition originelle est conservée, mais encadrée par des balises. Par exemple :

```
\let\t@gfbox=\fbox
➤  \def\fbox#1{%
     \xBegin{fbox}\t@gfbox{#1}\xEnd{fbox}}
```

- Technique de *simplification* : La macro est réécrite de façon simplifiée, en profitant du fait que la mise en page n'est pas conservée dans le XML.

```
\def\@maketitle{%
\@title%
➤  \@author%
\@date%
}
```

- *Balisage à façon*, quand il faut baliser des structures complexes : le point de départ est alors souvent les macros d'origines, qui sont alors plus ou moins

modifiées. C'est la méthode utilisée par exemple le balisage des tables dans `tabularx_lxir.sty`.

Vous pouvez vous inspirer des classes et styles déjà balisés. Si vous êtes un expert LaTeX, nul doute que vous pourrez faire beaucoup mieux !

4.1.3 Écrire un nouveau style en introduisant le balisage LXir

Si vous envisagez d'écrire une classe ou un style et que vous souhaitez utiliser LXir pour la conversion de vos documents en XML, vous pouvez inclure directement les macros de balisage définies dans lxirtag. Ces macros sont inactives tant que le style lxir n'est pas appelé, et seront donc sans effet en dehors du contexte de conversion vers XML avec LXir. Le style lxirdoc est un exemple simple de cette technique.

4.2 Modifier le XML obtenu

Partir du XML fourni par LXir est la façon la plus simple d'obtenir le XML que vous souhaitez. Vous pouvez par exemple écrire votre feuille de style XSLT et l'appliquer au XML LXir avec votre moteur XSLT favori. Vous pouvez aussi l'inclure dans la chaîne de transformation du programme *lxir*.

Plus généralement, vous pouvez modifier cette chaîne et accéder au XML produit à chacune des étapes permettant d'aboutir au XML final. Ces étapes sont décrites dans le fichier `transformations.xml`. Les transformations de types XSLT sont repérables par la valeur de l'attribut *name* (xslt_proc), par exemple :

```
➤   <transformation name="xslt_proc" param="clean.xsl"/>
```

Les autres transformations font appel à des traitements DOM inclus dans des fonctions C du programme *lxir*.

À chaque étape de la transformation, vous pouvez obtenir dans un fichier une copie de l'arbre XML en insérant un transformation de type *dump_tree* :

```
➤   <transformation name="dump_tree" param="debug_clean.xml"/>
```

Les étapes sont regroupées dans deux piles (les éléments « stack ») : les transformations permettant la construction des formules mathématiques en MathML sont listées dans l'élément « stack » de type « math », tandis que les autres transformations sont listées dans l'élément « stack » de type « text ».

Par exemple, si vous souhaitez ajouter votre transformation XSLT finale `mon_lxir2html.xsl` pour passer du « XML LXir » à du XHTML, vous devez ajouter un élément à la fin du « stack » de type « text » :

```
➤   <transformation name="xslt_proc" param="mon_lxir2html.xsl"/>
```

4.3 Ajouter une fonte non prise en compte

Si vous utilisez une fonte qui n'est pas prise en compte par LXir, vous pouvez ajouter les données de cette fonte dans un fichier XML et ajouter l'appel à ce fichier dans `config.xml`. Le fichier `zapfdingbats.xml`, fourni avec LXir, permet par exemple d'utiliser la fonte Zapf Dingbats dans vos documents.

5 Licence

LXir est développé sous licences libres :
- *LaTeX Project Public License (LPPL)* pour les fichiers intégrés à l'arborescence TEX ;
- *GNU Public License (GPL), version 3* pour tout le reste.

Références

[1] www.edpsciences.org

[2] www.cybertheses.org

[3] www.ajlsm.com

[4] Le site du convertisseur Hermes : `http://hermes.roua.org/`

[5] Le site de LaTeXML : `http://dlmf.nist.gov/LaTeXML/`

[6] Le site de Tralics : `http://www-sop.inria.fr/apics/tralics/`

[7] Sofka M.D., *TEX to HTML Translation via Tagged DVI Files*, TUGboat **19** (1998), p. 214.

[8] Le site du convertisseur tex4ht : `http://www.cse.ohio-state.edu/~gurari/TeX4ht/mn.html`