Structuration automatique en XML d'un dictionnaire électronique de l'indonésien à partir de documents Word

Yaying Liu, Damien Nouvel

Structuration automatique en XML d'un dictionnaire électronique de l'indonésien à partir de documents Word

Yaying LIU Damien NOUVEL
Inalco ERTIM
yingya621@gmail.com, damien.nouvel@inalco.fr

RÉSUMÉ
Les dictionnaires électroniques sont de plus en plus utilisés dans le contexte de la diffusion et de la popularisation des appareils électroniques et d'Internet. Dans ce contexte, la numérisation et la structuration des dictionnaires édités pour le format papier a tout intérêt à être réalisée. Le projet que nous présentons a pour objectif de convertir un dictionnaire indonésien, initialement rédigé sous Word, afin d'obtenir des bases de données sous forme de ressource lexicale.

MOTS-CLÉS : Dictionnaire électronique, Ressource lexicale, Indonésien.

KEYWORDS: Electronic dictionary, Lexical resource, Indonesian.

1 Présentation générale

De nos jours, avec les progrès technologiques informatiques et le niveau accru de la demande pour diverses ressources numériques, les usages de ces dernières dans le domaine des humanités numériques sont devenus de plus en plus fréquents et importants. La numérisation et la conversion des dictionnaires a indéniablement un effet positif sur l'enseignement, la diffusion et la préservation des langues. Elles fournissent également une importante source de données pour la recherche dans des domaines tels que la linguistique ou le Traitement Automatique des Langues (TAL). Cependant, un grand nombre de dictionnaires sont actuellement stockés dans des fichiers peu structurés, comme Word, qui permettent aux lexicographes de les éditer mais qui ne sont pas adaptées comme formats de données pour les dictionnaires électroniques (Joffe & De Schryver, 2004). Afin d'obtenir des données adaptées, il faut les convertir en un format structuré.

Il existe plusieurs méthodes de conversion, notamment la saisie manuelle des informations, la conversion à l'aide de langages informatiques et la conversion automatique par apprentissage automatique. Saisir manuellement les informations relatives aux articles qui décrivent les entrées du dictionnaire peut être réalisé avec un logiciel d'édition de dictionnaires, ce qui demande beaucoup de travail mais permet des conversions précises et de bonne qualité. L'approche par le programme informatique à base de règles permet d'économiser en temps de main-d'œuvre, mais ne permet pas d'atteindre un haut niveau de précision de conversion pour tous les articles. Les modèles d'apprentissage automatique, tels que le GROBID-dictionary (Khemakhem *et al.*, 2018), sont encore au stade expérimental, difficiles à mettre en œuvre en pratique.

Ce projet vise à construire un dictionnaire électronique pour le Dictionnaire Indonésien Français général (DIF), à l'aide des langages informatiques et des méthodes de TAL. L'étape actuelle du travail vise à terminer la conversion des ressources du dictionnaire en format Word en une base de données

électroniques valide, vérifiée et utilisable.

2 État de l'art

Dans cette section, nous présentons les différents processus de création d'un dictionnaire électronique. Nous commençons d'abord par la présentation des méthodes de conversion du format Word, puis, nous présentons les différents formalismes de structures des entrées. Pour terminer, nous parlons de la post-édition.

2.1 Conversion au format de stockage

Parmi les projets similaires au nôtre, le DiLAF (Enguehard & Mangeot, 2014) est un projet de dictionnaire dont les sources du dictionnaire ont été saisies sous Word, au format DOC. La première étape consiste à convertir les fichiers en DOCX à l'aide d'OpenOffice, puis à les décompresser pour obtenir un format XML. Ensuite, des des langages réguliers implémentés avec outils (plugin TexFx, Textwrangler, notepad++, etc.) permettent d'obtenir un format LMF (Lexical Mark-up Framework) comme modèle de base d'entrée du dictionnaire. Un autre projet similaire a porté sur un dictionnaire galicien (Guinovart & Simões, 2013). Les premières étapes sont similaires au précédent, afin d'obtenir un format XML. Par la suite, le module Text :RewriteRules de Perl permet de modifier les balises et les contenus du fichier XML. Une fois ces étapes de remplacement et de nettoyage terminées, des DTD sont utilisées pour construire la structure des entrées du dictionnaire. Avec les balises portant des informations textuelles, des expressions régulières sont utilisées pour remplacer les balises inutiles par de nouvelles balises.

Contrairement aux méthodes choisies pour ces deux projets, nous avons choisi une méthode de conversion différente pour notre projet. Tout d'abord, nous avons utilisé Python pour convertir les fichiers DOC en fichiers DOCX, puis nous avons utilisé le module Python « DOCX-Python », pour repérer les différentes polices et caractéristiques structurelles du contenus des entrées. Nous avons ensuite extrait directement les différentes structures des entrées qui sont stockées en mémoire selon une structure adaptée aux contenus extraits. Enfin, nous avons utilisé le module « Yattag » pour exporter ces contenus avec des balises correspondant aux différentes parties des entrées, afin de générer un fichier XML pour chaque fichier Word.

2.2 Formalismes de structuration d'entrées lexicographiques

Nous adoptons la norme TEI (Sperberg-McQueen *et al.*, 1994) qui est une normalisation bien établie qui s'est avérée populaire au sein de la communauté lexicographique. Les principaux éléments structurels du chapitre des dictionnaires TEI sont présentés dans la description de (Romary, 2013) :

— `<entry>` est l'élément structurant de base d'un lexique et regroupe les informations de forme, les informations grammaticales, les informations de sens et les renvois ;
— `<form>` peut être utilisé pour décrire une ou plusieurs formes associées à une entrée ;
— `<gramGrp>` regroupe toutes les caractéristiques grammaticales qui peuvent être attachées à l'entrée dans son ensemble (par le biais de son appartenance à la classe de modèle mo-

del.entryPart.top), à une forme spécifique (à travers la classe de modèle model.formPart) ou encore comme contrainte sur l'un des sens d'un mot (toujours à travers model.entryPart.top) ;
— `<sense>` rassemble toutes les informations relatives aux sens, c'est-à-dire les définitions, les exemples, l'usage et les informations d'utilisation et les notes supplémentaires.

TEI n'établit pas de modèle de base pour les entrées. Les quatre structures de base proposées par (Romary, 2013) sont basées sur l'analyse et le résumé de la structure de base de l'entrée. De plus, TEI permet de choisir entre différentes options d'encodage pour le même élément d'information, le lexique encodé dans TEI peut alors avoir des schémas différents.

D'autres formats lexicographiques existent, comme les structures LMF (Francopoulo *et al.*, 2006) et OntoLex-Lemon (McCrae *et al.*, 2017), qui proposent un modèle général comprenant le modèle de base générique et des extensions de ce modèle de base. Toutefois, les extensions de LMF doivent s'appuyer sur la partie *core* pour décrire les données, ce qui n'est pas explicitement souligné par le modèle OntoLex-Lemon. Diverses extensions de LMF mettent l'accent sur l'adaptation aux données du domaine TAL, tandis que OntoLex-Lemon met l'accent sur certaines caractéristiques des entrées dans le domaine des ontologies. Par exemple, dans le modèle OntoLex-Lemon le sens actuel d'un mot est donné par référence à un concept ontologique et les sens lexicaux ne représentent que la mise en correspondance d'un mot à un concept. LMF est moins flexible que TEI. Il permet plus de contrôle sur les pratiques d'encodage et fournit un formalisme plus contraignant pour la modélisation de l'information lexicale.

2.3 Post-édition

À l'heure actuelle, la façon dont est envisagée la création des bases de données pour les dictionnaires électroniques est d'utiliser un langage informatique pour créer et alimenter la base de données, mais cette méthode peut générer des contenus erronés, que l'ordinateur ne pourra pas traiter correctement. Il existe plusieurs manières de réaliser la post-édition. Par exemple, Jürviste *et al.* (2011) ont choisi EELex comme éditeur pour la post-édition de projet. Leur dictionnaire, EBD (« Basic Estonian Dictionary »), est un système interactif, conçu pour les apprenants estoniens (Kallas *et al.*, 2015). Enguehard & Mangeot (2014) ont choisi la plateforme Jibiki comme éditeur pour la poste-édition. Jibiki (Mangeot, 2002) est basé sur un modèle d'interface HTML instancié par l'entrée lexicale à publier. Au lieu de choisir directement des logiciels d'édition existants, Simões *et al.* (2016) ont choisi de créer leurs propres programmes de post-édition. Ils ont choisi eXiste-BD pour importer leur XML.

Pour notre projet, nous utilisons une autre plateforme d'édition du dictionnaire, Lexonomy (Měchura *et al.*, 2017). Cette plateforme open-source permet la rédaction et la publication de dictionnaires.

3 Description de l'indonésien et des sources

Comme la majorité des langues austronésiennes, l'indonésien est une langue agglutinante. L'une des caractéristiques les plus importantes est que la fonction grammaticale est exprimée par l'ajout de différents affixes au début ou à la fin des noms, des verbes, etc., ces informations étant présentes dans le dictionnaire. Dans cette langue, il y a trois grandes catégories morphologiques : les mots de base, les mot-outils et les affixes. Les mots de base peuvent fonctionner de façon autonome, mais

sont également susceptibles d'affixation. Ils peuvent aussi être redoublés (Sneddon *et al.*, 1996).

Le Dictionnaire indonésien français général (Labrousse, 1984) a été réalisé par Pierre Labrousse, professeur d'indonésien à l'Inalco. Le projet de numérisation du DIF a été initié par l'Inalco pour faire évoluer le dictionnaire papier en un dictionnaire électronique en ligne, afin de faciliter la recherche des entrées et le partage de cette ressource. Après cette édition papier du dictionnaire, P. Labrousse a décidé d'en réaliser une version électronique. Pour cela, il a rédigé les articles de cette nouvelle version du dictionnaire au format DOC, et l'a organisé en deux niveaux de structure : le niveau sémantique et le niveau lexical. Au niveau sémantique, les fichiers sont ordonnés par deux grandes sujets : « l'homme » et « la société ». Chaque grand sujet est organisé en des sous-sujets. Par exemple, dans le sujet « l'homme », il existe plusieurs sous-sujets tels que *usia* (l'âge), *tubuh* (le corps), *perasaan* (la perception), etc. Dans chaque fichier, les articles du même sujet sémantique sont ordonnés alphabétiquement.

Un article complet se compose de deux lignes, dont la première contient l'entrée, et souvent des informations historiques. La deuxième ligne contient toutes les informations relatives à l'entrée, notamment les formes des mots, les définitions, les exemples, les traductions, etc. Les différentes parties d'un article sont codées par utilisation de symboles, de polices, de formats.

@II. AZAM. [ar.]
n. intention f., propos m. vv. *niat, maksud, tujuan.* *** berazam. 1.** avoir une intention : *berazam berdikari* avoir l'intention d'être indépendant. vv. *berniat, bermaksud.* **2.** être résolu, déterminé. vv. *bertekad.* *** mengazamkan.** avoir l'intention de, faire tout ce qui est en son pouvoir pour : *mengazamkan hidupnya untuk berdakwah* consacrer sa vie à la propagation de la foi. $ *mengazamkan diri* se consacrer. vv. *meniatkan, memaksudkan.* *** keazaman.** volonté f., intention f. vv. *niat, maksud.*

FIGURE 1 – Exemple d'article de dictionnaire DIF

La figure 2 montre un article général du dictionnaire. Dans la première ligne, il y a deux blocs. Un bloc commence par le symbole « @ », suivi de la forme associée à l'entrée (mot vedette), toujours en gras et en lettres capitales. Le deuxième bloc est entre crochets. Dans la deuxième ligne, nous trouvons tour à tour des informations sur la catégorie, la définition, l'exemple, la traduction, la référence, les synonymes et des sous-entrées. Ces informations apparaissent en même temps que certains symboles ou chiffres spéciaux (« * », « vv. », « $ », etc.). De nombreux échanges avec l'auteur du dictionnaire ont permis de déterminer les parties d'un article.

4 Déroulement du projet

Dans cette section, nous présentons deux grandes parties. Dans la première partie, « Conversions », nous allons décrire les méthodes de deux différentes conversions. Dans la deuxième partie, nous présenterons le travail de post-édition avec le logiciel d'éditeur Lexonomy.

4.1 Conversions

Conversion DOC en DOCX

Constatant que les structures du fichier original sont marquées par leur propre format, nous utilisons le module Python-DOCX de Python qui peut traiter le contenu du fichier DOCX en fonction des différentes caractéristiques de format. Il suffit de convertir au préalable le fichier DOC en fichier DOCX, puis d'analyser ce fichier DOCX. Avec le module *pywin32* de Python, nous avons converti les fichiers DOC en DOCX.

Conversion DOCX en XML

Comme il s'agit d'une étape cruciale de tout le projet, nous décrivons en détail le processus de conversion des articles. La première étape de ce processus est l'extraction et la deuxième étape est l'ajout de balises XML. La logique de ce processus est d'extraire séparément les différents éléments des entrées et d'y ajouter les balises appropriées, les formats ainsi en des entrées qui répondent aux demandes du stockage électronique.

Extraction

Après avoir obtenu le fichier DOCX nécessaire pour la session d'extraction, nous analysons la structure des articles afin d'implémenter un algorithme, sous forme d'un programme Python. Nous avons choisi une méthode d'extraction qui peut extraire et créer une structure assez générique et complète. Elle comprend les contenus tels que l'entrée, la fenêtre historique, plusieurs définitions, les exemples indonésiens, leurs traductions françaises, la partie de forme affixée et la partie de mot composé, comme le montre la figure 2.

Dans cet exemple, nous divisons l'article en six parties : l'entrée, la fenêtre historique, la catégorie, les définitions, la partie de forme affixée et la partie de mot composé.

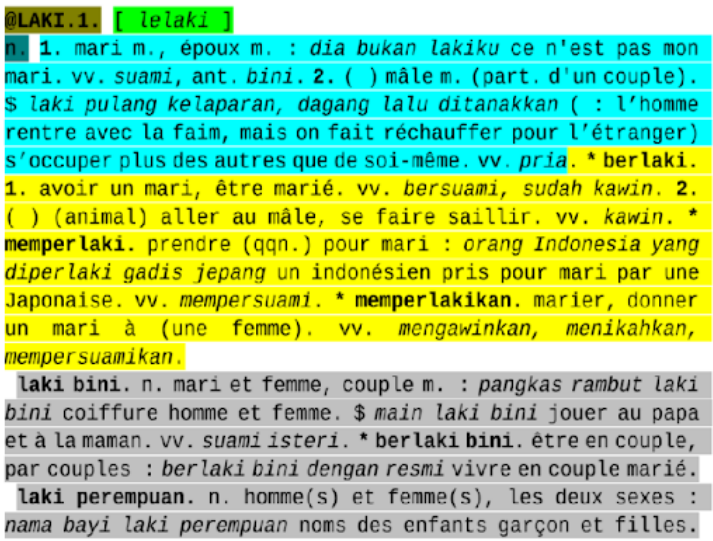

FIGURE 2 – Exemple d'article divisé en six parties

Algorithme du programme d'extraction

La logique de l'algorithme du programme d'extraction est basée sur les informations relatives à la structure d'article, construite en utilisant différents symboles spéciaux et formats textuels, illustrée

par le schéma 3. Au préalable, le programme doit éviter les renvois, qui commencent par « @ » et « < > », dans ce cas le programme n'en tient pas compte. Sinon, si la partie entrée contient le symbole « @ » (sans chevrons) et des lettres en gras (run.bold pour Python-DOCX), le programme peut démarrer l'extraction d'une entrée lexicale.

La figure 2 présente la structure principale d'une entrée, qui contiendra au maximum six éléments à extraire. Le premier élément est le mot vedette, en majuscules. Nous déterminons ensuite si l'entrée contient une fenêtre historique, en testant la présence de crochets « [] », et en utilisant le cas échéant une expression régulière pour la capturer. Ensuite, le programme recherche la présence d'une catégorie lexicale à l'aide d'une liste des catégories lexicales possibles. Une entrée commence par une définition, un exemple en italique et sa traduction française au format *normal*, ces parties sont enregistrées. De même, il repère ensuite les affixes, en gras et suivis d'une étoile « * », les mots composés, en gras mais sans « * ». Notons qu'une entrée peut contenir plusieurs définitions (et traductions associées), qui sont alors numérotées. Toutes ces caractéristiques de format nous permettent de les détecter dans une très large majorité des cas. À la fin de chaque définition peuvent apparaître la référence, le synonyme et l'expression composée, qui sont indiqués respectivement par les signes « vv. », « # », et « ♢ » repérées et extraites par langages réguliers.

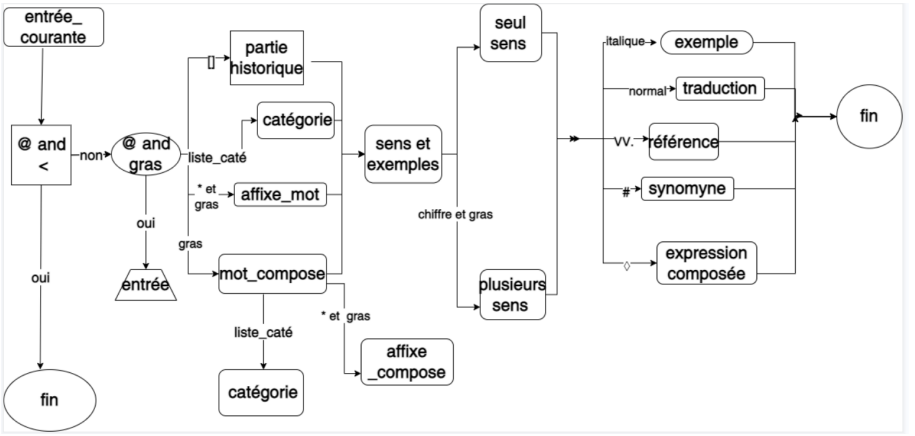

FIGURE 3 – Schéma d'extraction

Génération du format XML

Dans notre processus d'extraction de la structure, tous les articles sont extraits dans des dictionnaires en mémoire afin de produire des entrées au format XML. Nous utilisons la librairie Python Yattag pour faciliter la génération du XML et rendre plus lisible l'organisation de la structure du fichier en sortie. La figure 4 montre un exemple d'article dans un fichier XML.

```xml
<sense n="1">
    <def> placenta m. :</def>
    <exemples>
        <cite type="exemple" xml:lang="id">
            <quote>penguburan ari-ari</quote>
        </cite>
        <cite type="translation" xml:lang="fr">
            <quote>enterrement du placenta. $</quote>
        </cite>
        <cite type="exemple" xml:lang="id">
            <quote>(exp)(tali) ariarikemudian ariari dipotong</quote>
        </cite>
        <cite type="translation" xml:lang="fr">
            <quote>cordon ombilical m. :ensuite on coupe le cordon ombilical. vv..</quote>
        </cite>
    </exemples>
    <xr type="reference">
        <ref>tembuni, bali, uri, plasénta</ref>
    </xr>
    <xr type="syn">
        <ref></ref>
    </xr>
    <xr type="exp_compose">
        <ref></ref>
    </xr>
</sense>
```

FIGURE 4 – Une partie d'entrée sous TEI dans un fichier XML

4.2 Lexonomy (post-édition)

Lexonomy [1] est une plateforme en ligne pour l'écriture et la publication de dictionnaires. Sa mission est de fournir un outil facile à utiliser pour les petits et moyens projets de dictionnaires. Sur cette plateforme, nous pouvons mettre en œuvre les trois principales fonctions d'importation, de modification et d'exportation. La figure 5 montre l'interface de la plateforme pour modifier une entrée, qui permet déditer, d'ajouter ou de supprimer des nœuds XML avec une interface agréable à utiliser et facile à prendre en main.

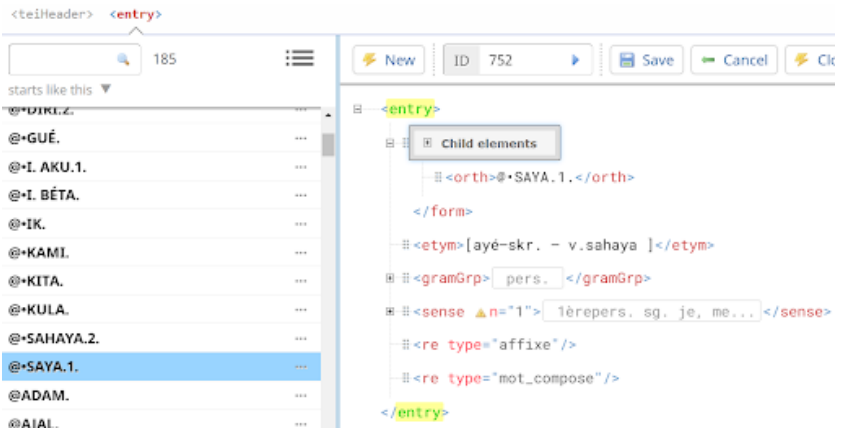

FIGURE 5 – Capture d'interface de la modification de Lexonomy

1. https ://www.lexonomy.eu/

5 Résultats et discussion

Bien entendu, le résultat final du projet dépend directement de la précision de la partie extraction, puisque la partie ultérieure du processus d'ajout de balises est entièrement basée sur les balises du dictionnaire extraites dans les contenus des entrées formatées sous Word. Le modèle d'extraction existant est un modèle générique et implémenté incrémentalement, qui a été construit pour la plupart des structures d'entrées.

Sur un échantillon de 329 articles du DIF, nous calculons que notre programme peut traiter 89,04% des articles, ce qui nous épargne beaucoup de travail manuel. Analysons maintenant le fichier XML final obtenu par ce processus de conversion.

Pour vérifier l'extraction, 16 fichiers sur deux sujets distincts ont été vérifiés manuellement. Le sujet « homme » produit 6 fichiers XML contenant 118 entrées lexicales. Le sujet « âge » génère 10 fichiers XML, qui contiennent 157 entrées extraites correctement par le programme (dont certaines résultent de mots composés et des formes affixées, traitées comme des entrées individuelles).

L'analyse de ces résultats nous a permis de conclure que le programme existant était capable d'extraire une grande partie du contenu automatiquement, avec une précision satisfaisante. Concernant les erreurs, deux raisons principales ont été identifiées. La première concerne les erreurs humaines liées à une utilisation erronée des formats dans le fichier source (par exemple des mots composés précédés de « @ », ce qui n'est pas prévu. La seconde est liée à l'incomplétude du programme : comme il n'est pas possible d'analyser la structure de toutes les entrées, en particulier la partie concernant sens, le programme existant est incomplet et n'est pas en mesure d'extraire le contenu avec précision face à certaines structures trop spécifiques et inattendues.

Bien que le fichier XML existant comporte quelques erreurs et nécessite une session de relecture ultérieure, le mode de conversion automatique réduit beaucoup le travail humain. En plus, dans ce mode de transformation détourné, toutes les entrées sont décomposées et reconstruites, ce qui facilite leur utilisation sous forme électronique (consultation, site web, outils TAL, etc.). Parce que tous les contenus requis de l'article sont extraits et stockés séparément, les lexicographes pourront alors éditer les entrées et reconstruire la structure des articles, sans tenir compte des contraintes de mise en page et de lisibilité des dictionnaires papier.

6 Conclusion

Le travail décrit dans cet article porte sur la conversion d'un dictionnaire de l'indonésien depuis sont format actuel sous Word vers un format XML structuré et exploitable. Pour ce faire, un programme de conversion a été implémenté, qui s'appuie sur les formats (symboles, gras, italique, etc.) du fichier source afin d'extraire les informations utiles en mémoire et de produire un fichier XML contenant les entrées structurées selon un format TEI. Le programme donne des résultats satisfaisants pour une tâche qui serait autrement fastidieuse. Les fichiers XML résultants sont déposés sur une plateforme d'édition de dictionnaire, Lexonomy, qui facilitera leur post-édition par les auteurs de dictionnaires et pourra permettre de publier le dictionnaire et de rechercher dans ses entrées.

Références

ENGUEHARD C. & MANGEOT M. (2014). Computerization of african languages-french dictionaries. *arXiv preprint arXiv :1405.5893*.

FRANCOPOULO G., GEORGE M., CALZOLARI N., MONACHINI M., BEL N., PET M. & SORIA C. (2006). Lexical markup framework (lmf). In *International Conference on Language Resources and Evaluation-LREC 2006*.

GUINOVART X. G. & SIMÕES A. (2013). Retreading dictionaries for the 21st century. In *2nd Symposium on languages, applications and technologies* : Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

JOFFE D. & DE SCHRYVER G.-M. (2004). Tshwanelex : A state-of-the-art dictionary compilation program. In *11th EURALEX International Congress (EURALEX-2004)*, p. 99–104 : Faculté des Lettres et des Sciences Humaines.

JÜRVISTE M., KALLAS J., LANGEMETS M., TUULIK M. & VIKS Ü. (2011). Extending the functions of the eelex dictionary writing system using the example of the basic estonian dictionary. *eLexicography in the 21st Century : New Applications for New Users, Proceedings of eLex*, p. 106–112.

KALLAS J., KILGARRIFF A., KOPPEL K., KUDRITSKI E., LANGEMETS M., MICHELFEIT J., TUULIK M. & VIKS Ü. (2015). Automatic generation of the estonian collocations dictionary database. In *Electronic lexicography in the 21st century : linking lexical data in the digital age. Proceedings of the eLex 2015 conference*, p. 11–13.

KHEMAKHEM M., HEROLD A. & ROMARY L. (2018). Enhancing usability for automatically structuring digitised dictionaries. In *GLOBALEX workshop at LREC 2018*.

LABROUSSE P. (1984). *Dictionnaire général : Indonésien-français*. Cahiers d'Archipel. Éditions de la Maison des sciences de l'homme, Paris.

MANGEOT M. (2002). Projet papillon : intégration de dictionnaires existants et gestion des contributions. In *JST'02 Journées Science et Technologie*, p. 64–65.

MCCRAE J. P., BOSQUE-GIL J., GRACIA J., BUITELAAR P. & CIMIANO P. (2017). The ontolex-lemon model : development and applications. In *Proceedings of eLex 2017 conference*, p. 19–21.

MĚCHURA M. B. *et al.* (2017). Introducing lexonomy : an open-source dictionary writing and publishing system. In *Electronic Lexicography in the 21st Century : Lexicography from Scratch. Proceedings of the eLex 2017 conference*, p. 19–21.

ROMARY L. (2013). Tei and lmf crosswalks. *arXiv preprint arXiv :1301.2444*.

SIMÕES A., ALMEIDA J. J. & SALGADO A. (2016). Building a dictionary using xml technology. In *5th Symposium on Languages, Applications and Technologies (SLATE'16)* : Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

SNEDDON J., AUSTRALIA. COMMONWEALTH DEPT. OF EMPLOYMENT E., TRAINING & EWING M. (1996). *Indonesian : A Comprehensive Grammar*. Comprehensive grammars. Routledge.

SPERBERG-MCQUEEN C., FOR COMPUTERS A. & THE HUMANITIES (1994). *Guidelines for Electronic Text Encoding and Interchange*. Electronic book library. Text Incoding Initiative.