

ÉCOLE NATIONALE DES CHARTES

Caroline Corbières

licenciée ès lettres

Du catalogue au fichier TEI

**Création d'un workflow pour encoder
automatiquement en XML-TEI des
catalogues d'exposition**

Mémoire pour le diplôme de master
« Technologies numériques appliquées à l'histoire »

2020

Résumé

Le présent mémoire a été rédigé à la suite d'un stage de quatre mois au sein du projet Artl@s dirigé par Béatrice Joyeux-Prunel et financé en partie par l'École normale supérieure et le centre IMAGO, centre d'excellence Jean Monnet. Ce projet de recherche en histoire de l'art et en humanités numériques vise à rassembler les données des catalogues d'exposition des XIX^e et XX^e siècle issus du monde entier dans la base de données BasArt. L'objectif du stage a été de proposer un *workflow* s'appuyant sur le précédent travail de Lucie Rondeau du Noyer afin d'automatiser la récupération des entrées des catalogues pour les verser dans BasArt. Le *workflow* permet de traiter un catalogue de sa forme numérique (PDF ou JPEG) jusqu'à son encodage en XML-TEI. Dans un second temps, il a été question de documenter la chaîne de traitement afin de prévoir son réemploi par les membres du projet. Ce mémoire s'attache à analyser les enjeux de la réalisation d'un tel *workflow* à travers l'intégration des fichiers ALTO dans le processus de transformation des fichiers et la prise en compte des besoins des membres d'Artl@s.

Mots-clés : ALTO ; Artl@s ; catalogues d'exposition ; GROBID-dictionaries ; histoire de l'art ; numérique ; OCR ; XML-TEI ; XSL-FO ; workflow.

Informations bibliographiques : Caroline Corbières, *Du catalogue au fichier TEI : Création d'un workflow pour encoder automatiquement en XML-TEI des catalogues d'exposition*, mémoire de master « Technologies numériques appliquées à l'histoire », dir. Thibault Clérice et Béatrice Joyeux-Prunel, École nationale des chartes, 2020.

Remerciements

Je souhaite tout d'abord remercier ma tutrice de stage, Béatrice Joyeux-Prunel, pour sa confiance, ses conseils mais également son adaptabilité face à la crise sanitaire.

Je remercie aussi Thibault Clérice, mon directeur de mémoire, pour ses recommandations, notamment lorsqu'il a fallu trouver des solutions pour combler le retard de la livraison de l'application autour de laquelle ce stage s'est construit.

Je tiens également à adresser de sincères remerciements à Simon Gabay qui m'a suivie tout au long de ce stage. Nos échanges presque quotidiens m'ont permis d'avancer malgré les aléas des logiciels informatiques, du projet et du télétravail.

Je remercie de même mes collègues Ljudmila Petkovic, Auriane Quoix et Barbara Topalov sans qui certaines étapes de ce stage n'auraient pas pu être réalisées. J'adresse également mes remerciements à Lucie Rondeau du Noyer, Mohamed Khemakhem, Laurent Romary, Jean-Paul Rehr et Matthias Gille Lenvenson pour leur aide précieuse, tant par leurs conseils que par leur travail.

J'adresse de chaleureux remerciements à mes cher·e·s ami·e·s Agathe et Alexandre, sans qui le master n'aurait pas eu la même saveur. Enfin, merci infiniment à ma famille et à Nathan pour leurs relectures et leur soutien sans faille.

Bibliographie

Sources : catalogues d'exposition

- Catalogue de l'exposition des œuvres de Claude Monet. 9 Boulevard de la Madeleine.*
Ouverte du 1er au 25 mars [1883], Paris, 1883.
- Catalogue illustré de la section japonaise à l'exposition internationale des arts décoratifs et industriels modernes. Paris.* 1925, Paris, 1925.
- Exposition Charles Cottet : catalogue*, avec la coll. de Léonce Bénédite, 1911.
- Exposition des œuvres de Gustave Courbet à l'école des Beaux-Arts (mai 1882)*, avec la coll. de Jules-Antoine Castagnary, Paris, 1882.
- LIOU (Té-Péou), *Exposition chinoise d'art ancien et moderne / organisée sous le patronage du commissariat général de la République à Strasbourg et du Ministre plénipotentiaire de Chine en France, Palais du Rhin, mai-juillet 1924*, avec la coll. de Tsai Yen-Pei, Strasbourg, 1924.
- MONTROSS GALLERY et MATISSE (Henri), *Henri Matisse Exhibition, January 20th to February 27, 1915, catalogue*, New York, 1915.
- II bienal de São Paulo, Museu de arte moderna*, dir. Museu de arte moderna et Biennale internationale de São Paulo, São Paulo, Brésil, 1953.
- SALON DES INDÉPENDANTS, *Société des artistes indépendants. 8, Catalogue des œuvres exposées : [8e Exposition du 19 mars au 27 avril 1892, Pavillon de la ville de Paris, Champs Elysées] / Société des artistes indépendants*, Paris, 1892.
- *Société des artistes indépendants. 22, Catalogue de la 22e exposition 1906 : grandes serres de la ville de Paris... du 20 mars au 30 avril... / Société des artistes indépendants*, dir. L'Emancipatrice, Paris, 1906.
- *Société des artistes indépendants. 29, Catalogue de la 29e exposition, 1913 : Quai d'Orsay, Pont de l'Alma, du 19 mars au 18 mai inclus / Société des artistes indépendants*, Paris, 1913.
- *Société des artistes indépendants. 34, Catalogue de la 34e exposition 1923 : Grand Palais des Champs-Elysées, du 10 février au 11 mars / Société des artistes indépendants*, Paris, 1923.
- *Société des artistes indépendants. 37, Catalogue de la 37e exposition 1926... du 20 mars au 2 mai / Société des artistes indépendants*, Paris, 1926.

- SOCIÉTÉ DES AMIS DES ARTS, *Catalogue de l'exposition des œuvres d'art d'artistes vivants : Strasbourg, Hôtel-de-ville, du 11 mai au 8 juin 1884*, Strasbourg, 1884.
- SOCIÉTÉ DES ARTISTES FRANÇAIS, *Catalogue illustré du Salon de 1887*, réd. par F.-G. Dumas, Paris, 1887.
- SOCIÉTÉ LORRAINE DES AMIS DES ARTS, *Catalogue des peintures, miniatures, aquarelles, dessins, sculptures et lithographies exposés à Nancy... par les artistes lorrains*, Nancy, 1847.
- The exhibition of the Royal Academy of Arts. MDCCCLIX. (1859). The ninety-first.* London, 1859.
- TOLEDO MUSEUM OF ART, *Catalogue of the inaugural exhibition, January seventeenth to February twelfth An. Dni. MCMXII*, avec la coll. d'Harold B. Lee, New York, 1912.
- VI. *Esposizione Internazionale d'Arte Della Città Di Venezia, 1905 : Catalogo Illustrato*, Venezia, 1905.

Histoire des catalogues

- BARBIER (Frédéric), DUBOIS (Thierry), SORDET (Yann) et BROGLIE (Gabriel de), *De l'argile au nuage, une archéologie des catalogues : IIe millénaire av. J. C. - XXIe siècle [ouvrage publié à l'occasion des expositions organisées par la Bibliothèque Mazarine & la Bibliothèque de Genève, Paris 13 mars - 15 mai 2015, Genève 18 septembre - 21 novembre 2015]*, avec la coll. de Bibliothèque Mazarine et Bibliothèque publique et universitaire, [Paris] Genève, 2015.
- FALGUIÈRES (Patricia), RECHT (Roland) et JUBERT (Roxane), *Cahiers du Musée national d'art moderne, Du catalogue*, Paris, France, 1996.
- HURLEY (Cecilia) et BARBILLON (Claire), *Le catalogue dans tous ses états : actes du colloque [Rencontres de l'École du Louvre], 12, 13 et 14 décembre 2012*, avec la coll. de Rencontres de l'École du Louvre et Institut national d'histoire de l'art, Paris, 2015 (Rencontres de l'École du Louvre).
- JOUFFRET (Jean) et POULAIN (Martine), « Les collections de catalogues de vente d'œuvres d'art dans les bibliothèques », *Les Nouvelles de l'INHA*-33 (déc. 2008), p. 18-19.
- JOYEUX-PRUNEL (Béatrice) et MARCEL (Olivier), « Exhibition Catalogues in the Globalization of Art. A Source for Social and Spatial Art History », *Artl@s Bulletin*, 4-2 (2015), p. 80-104.

Histoire, histoire de l'art et humanités numériques

- CLAVERT (Frédéric), « Une histoire par les données ? Le futur très proche de l'histoire des relations internationales », *Bulletin de l'Institut Pierre Renouvin*, N° 44-2 (1^{er} nov.

- 2016), p. 119-130, URL : <https://www.cairn.info/revue-bulletin-de-l-institut-pierre-renouvin-2016-2-page-119.htm> (visité le 08/08/2020).
- COURTIN (Antoine), *Retour sur la datavisualisation « Sur la piste des ventes d'antiques »*, Numérique et recherche en histoire de l'art, URL : <https://numrha.hypotheses.org/743> (visité le 30/07/2020).
- CUADRA (Ruth) et MICHELS (Suzanne), *Publishing German Sales, A Look under the Hood of the Getty Provenance Index*, The Getty Iris, 17 avr. 2013, URL : <https://blogs.getty.edu/iris/publishing-german-sales-a-look-under-the-hood-of-the-getty-provenance-index/> (visité le 04/08/2020).
- DOSSIN (Catherine), « Towards a Spatial (Digital) Art History », *Artl@s Bulletin*, 4-1 (11 juin 2015), URL : <https://docs.lib.purdue.edu/artlas/vol4/iss1/1>.
- DOSSIN (Catherine) et ALKEMA (Hanna), « Women Artists Shows · Salons · Societies : Towards a Global History of All-Women Exhibitions », *Artl@s Bulletin*, 8-1 (2019).
- LAUG (Patrick) et BOROUCHAKI (Houman), *The BL2D Mesh Generator : Beginner's Guide, User's and Programmer's Manual*, Research Report, INRIA, 1996, p. 48.
- LEMERCIER (Claire) et JOYEUX-PRUNEL (Béatrice), « Créer une base de données en histoire de l'art : comment s'y prendre ? », dans *L'Art et la Mesure. Histoire de l'art et méthodes quantitatives*, dir. Béatrice Joyeux-Prunel, 2010, p. 165-180.
- NELSON (Brent), « Curating Object-Oriented Collections Using the TEI », *Journal of the Text Encoding Initiative*-9 (24 sept. 2016), URL : <http://journals.openedition.org/jtei/1680> (visité le 04/08/2020).
- NOËL (Denise), « Les femmes peintres dans la seconde moitié du XIXe siècle », *Clio. Femmes, Genre, Histoire*-19 (1^{er} avr. 2004), Number : 19 Publisher : Éditions Belin, DOI : 10.4000/clio.646.
- SAINTE-RAYMOND (Lea) et COURTIN (Antoine), « Enriching and Cutting : How to Visualize Networks Thanks to Linked Open Data Platforms. » *Artl@s Bulletin*, 6-3 (30 nov. 2017), URL : <https://docs.lib.purdue.edu/artlas/vol6/iss3/7>.

Artl@s et Visual Contagions

- JOYEUX-PRUNEL (Béatrice), « Bases de données et gestion de projets en humanités numériques Les dessous du projet Artl@s », *Biens symboliques / Symbolic Goods* (, 12 avr. 2018), URL : <https://www.biens-symboliques.net/242> (visité le 30/07/2020).
- « Visual Contagions, the Art Historian, and the Digital Strategies to Work on Them », *Artl@s Bulletin*, 8-3 (31 déc. 2019), URL : <https://docs.lib.purdue.edu/artlas/vol8/iss3/8>.
 - « ARTL@S : A Spatial and Trans-national Art History Origins and Positions », *Artl@s Bulletin*, 1-1 (), URL : <https://docs.lib.purdue.edu/artlas/vol1/iss1/1/> (visité le 30/07/2020).

- JOYEUX-PRUNEL (Béatrice), DOSSIN (Catherine) et SAINT-RAYMOND (Léa), *Artl@s, Atl@s*, URL : <https://artlas.huma-num.fr/fr/> (visité le 30/07/2020).
- JOYEUX-PRUNEL (Béatrice), CHATONSKY (Grégory), SAINT-RAYMOND (Léa), GUICHARD (Charlotte) et BURKI (Marie José), *Visual Contagions*, Imago, URL : <https://www.imago.ens.fr/> (visité le 30/07/2020).
- *Visual Contagions*, GitLab, URL : <https://gitlab.unige.ch/Beatrice.Joyeux-Prunel/visual-contagions> (visité le 10/09/2020).
- MATEI (Sorin), « ARTL@S and BasArt : A Loose Coupling Strategy for Digital Humanities », *Artl@s Bulletin*, 1–1 (15 sept. 2012), URL : <https://docs.lib.psu.edu/artlas/vol1/iss1/2>.

Encoder automatiquement des documents en XML-TEI

- BOHBOT (Hervé), FRONTINI (Francesca), LUXARDO (Giancarlo), KHEMAKHEM (Mohamed) et ROMARY (Laurent), « Presenting the Nénufar Project : a Diachronic Digital Edition of the Petit Larousse Illustré », dans *GLOBALEX 2018 - Globalex workshop at LREC 2018*, Miyazaki, 2018, p. 1-6, URL : <https://hal.archives-ouvertes.fr/hal-01728328> (visité le 04/08/2020).
- BOWERS (Jack), KHEMAKHEM (Mohamed) et ROMARY (Laurent), « TEI Encoding of a Classical Mixtec Dictionary Using GROBID- Dictionaries », dans *ELEX 2019 : Smart Lexicography*, Sintra, 2019, URL : <https://hal.inria.fr/hal-02264033> (visité le 01/08/2020).
- Docker*, URL : <https://www.docker.com/> (visité le 27/09/2020).
- GABAY (Simon), RONDEAU DU NOYER (Lucie) et KHEMAKHEM (Mohamed), « Selling autograph manuscripts in 19th c. Paris : digitising the Revue des Autographes », dans *IX Convegno AIUCD*, Milan, 2020, URL : <https://hal.archives-ouvertes.fr/hal-02388407> (visité le 04/08/2020).
- GABAY (Simon), RONDEAU DU NOYER (Lucie), GILLE LEVENSON (Matthias), PETKOVIC (Ljudmila) et BARTZ (Alexandre), « Quantifying the Unknown : How many manuscripts of the marquise de Sévigné still exist ? », dans *Digital Humanities DH2020*, Ottawa, 2020 (DH2020 Book of Abstracts), URL : <https://hal.archives-ouvertes.fr/hal-02898929> (visité le 04/08/2020).
- *Katabase*, GitHub, URL : <https://github.com/katabase> (visité le 30/07/2020).
- KHEMAKHEM (Mohamed), *Grobid-dictionaries*, 29 juil. 2020, URL : <https://github.com/MedKhem/grobid-dictionaries> (visité le 30/07/2020).
- *GROBID-Dictionaries Documentation*, GitHub, URL : <https://grobid-dictionaries.readthedocs.io/en/latest/> (visité le 30/07/2020).

KHEMAKHEM (Mohamed), FOPPIANO (Luca) et ROMARY (Laurent), « Automatic Extraction of TEI Structures in Digitized Lexical Resources using Conditional Random Fields », dans *electronic lexicography, eLex 2017*, Leiden, 2017, URL : <https://hal.archives-ouvertes.fr/hal-01508868> (visité le 08/05/2020).

KHEMAKHEM (Mohamed), HEROLD (Axel) et ROMARY (Laurent), « Enhancing Usability for Automatically Structuring Digitised Dictionaries », dans *GLOBALEX workshop at LREC 2018*, Miyazaki, 2018, URL : <https://hal.archives-ouvertes.fr/hal-01708137> (visité le 08/05/2020).

KHEMAKHEM (Mohamed), ROMARY (Laurent), GABAY (Simon), BOHBOT (Hervé), FRONTINI (Francesca) et LUXARDO (Giancarlo), « Automatically Encoding Encyclopedic-like Resources in TEI », dans *The annual TEI Conference and Members Meeting*, Tokyo, Japan, 2018, URL : <https://hal.inria.fr/hal-01819505> (visité le 04/08/2020).

- « Fueling Time Machine : Information Extraction from Retro-Digitised Address Directories », dans *JADH2018 "Leveraging Open Data"*, Tokyo, 2018, URL : <https://hal.archives-ouvertes.fr/hal-01814189> (visité le 04/08/2020).
- « How OCR Performance can Impact on the Automatic Extraction of Dictionary Content Structures », dans *19th annual Conference and Members' Meeting of the Text Encoding Initiative Consortium (TEI) -What is text, really ? TEI and beyond*, Graz, 2019, URL : <https://hal.archives-ouvertes.fr/hal-02263276> (visité le 01/08/2020).
- « Information Extraction Workflow for Digitised Entry-based Documents », dans *DARIAH Annual event 2020*, Zagreb, 2020, URL : <https://hal.archives-ouvertes.fr/hal-02508549> (visité le 30/07/2020).

LINDEMANN (David), KHEMAKHEM (Mohamed) et ROMARY (Laurent), « Retro-digitizing and Automatically Structuring a Large Bibliography Collection », dans *European Association for Digital Humanities (EADH) Conference*, Galway, 2018, URL : <https://hal.archives-ouvertes.fr/hal-01941534> (visité le 04/08/2020).

PDF Command Line Tools, URL : <https://www.coherentpdf.com/> (visité le 27/09/2020).

ROMARY (Laurent) et LOPEZ (Patrice), « GROBID - Information Extraction from Scientific Publications », *ERCIM News*, 100 (janv. 2015), URL : <https://hal.inria.fr/hal-01673305> (visité le 04/08/2020).

RONDEAU DU NOYER (Lucie), *Encoder automatiquement des catalogues en XML-TEI. Principes, évaluation et application à la Revue des autographes de la librairie Charavay*, mémoire de master « Technologies numériques appliquées à l'histoire », dir, Thibault Clérice et Simon Gabay, Paris, École nationale des chartes, 2019.

RONDEAU DU NOYER (Lucie), GABAY (Simon), KHEMAKHEM (Mohamed) et ROMARY (Laurent), « Scaling up Automatic Structuring of Manuscript Sales Catalogues », dans *TEI 2019 : What is text, really ? TEI and beyond*, Graz, 2019, URL : <https://hal.inria.fr/hal-02272962> (visité le 04/08/2020).

TOPALOV (Barbara), GABAY (Simon), JOYEUX-PRUNEL (Béatrice), ROMARY (Laurent) et RONDEAU DU NOYER (Lucie), *Automating Artl@s - extracting data from exhibition catalogues*, 2020.

Machine learning

PUSTEJOVSKY (James) et STUBBS (Amber), *Natural language annotation for machine learning*, Sebastopol, CA, Etats-Unis d'Amérique, 2013.

ZHENG (Alice) et CASARI (Amanda), *Feature engineering for machine learning : principles and techniques for data scientists*, Beijing etc., Chine, Pays multiples, 2018.

Formats XML et transformations XSL

ALTO : Technical Metadata for Layout and Text Objects (Standards, Library of Congress), URL : <https://www.loc.gov/standards/alto/> (visité le 30/07/2020).

BELAÏD (Abdel), RANGONI (Yves) et FALK (Ingrid), « Représentation des données en XML pour l'analyse d'images de documents », dans *CIDE 10*, Nancy, 2007, URL : <http://lodel.irevues.inist.fr/cide/index.php?id=147#tocto2n1> (visité le 09/08/2020).

BURNARD (Lou), *Qu'est-ce que la Text Encoding Initiative ?*, Marseille, 2015.

CONSORTIUM (TEI), *TEI P5 : Guidelines for Electronic Text Encoding and Interchange*, 13 févr. 2020, URL : <https://zenodo.org/record/3413524> (visité le 30/07/2020).

Extensible Stylesheet Language (XSL) Version 1.1, URL : <https://www.w3.org/TR/xsl/#d0e4611> (visité le 30/07/2020).

lxml - Processing XML and HTML with Python, URL : <https://lxml.de/> (visité le 27/09/2020).

RENDERX, *RenderX - Support - XSL Formatting Objects Tutorial - RenderX*, URL : <http://www.renderx.com/tutorial.html> (visité le 30/07/2020).

— *XEP User Guide - Java XML to PDF, PostScript XSL-FO Formatter < Support < XML to PDF, PostScript, AFP, Print - RenderX*, URL : <http://www.renderx.com/reference.html> (visité le 30/07/2020).

ReportLab - Content to PDF Solutions, URL : <https://www.reportlab.com/> (visité le 27/09/2020).

Techniques et formats de conversion en mode texte, BnF, URL : <https://www.bnf.fr/fr/techniques-et-formats-de-conversion-en-mode-texte> (visité le 09/08/2020).

ZARNDT (Frederick), BAUER (Joachim), ENDERS (Markus), GEIGER (Brian), HOCK (Kia Siang) et KERVINEN (Jukka), « The ALTO Editorial Board : Collaboration and Cooperation across Borders », dans *IFLA World Library and Information Congress*, Singapore, 2013.

OCR

CHASSINET (Philippe) et PERRIN (Emmanuelle), *Rendre visible la face cachée de l'iceberg*, ArchéOrient - Le Blog, URL : <https://archeorient.hypotheses.org/14807> (visité le 30/07/2020).

CLAUSNER (Christian), ANTONACOPOULOS (Apostolos) et PLETSCHACHER (Stefan), « Efficient and effective OCR engine training », *International Journal on Document Analysis and Recognition (IJDAR)* (, 9 oct. 2019), DOI : 10.1007/s10032-019-00347-8.

GABAY (Simon) et PETKOVIC (Ljudmila), *19th fixed-price and auction catalogues : Ground Truth and Models for OCR*, GitHub, URL : <https://github.com/ljpetkovic/OCR-cat> (visité le 30/07/2020).

KIESSLING (Benjamin), *Kraken - an Universal Text Recognizer for the Humanities*, URL : <https://dev.clariah.nl/files/dh2019/boa/0673.html> (visité le 04/08/2020).

MITTAGESSEN, *kraken*, original-date : 2015-05-19T09:24:38Z, 23 sept. 2020, URL : <https://github.com/mittagessen/kraken> (visité le 23/09/2020).

OLIVEIRA (Sofia Ares), SEGUIN (Benoit) et KAPLAN (Frederic), « dhSegment : A generic deep-learning approach for document segmentation », *2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR)* (, août 2018), p. 7-12, DOI : 10.1109/ICFHR-2018.2018.00011.

PERRIN (Emmanuelle), *Transkribus Tutorial*, nov. 2019, URL : <https://hal.archives-ouvertes.fr/hal-02472234> (visité le 30/07/2020).

SCHLAGDENHAUFFEN (Régis), *Optical Recognition Assisted Transcription with Transkribus : The Experiment concerning Eugène Wilhelm's Personal Diary (1885-1951)*, mars 2020, URL : <https://hal.archives-ouvertes.fr/hal-02520508> (visité le 30/07/2020).

Open source

ALLEN (Jonathan P.), « Three Strategies for Open Source Deployment : Substitution, Innovation, and Knowledge Reuse », dans *6th International IFIP WG 2.13 Conference on Open Source Systems (OSS)*, dir. Pär {\textbackslash}AAgerfalk ; Cornelia Boldyreff ; Jesús M. González-Barahona ; Gregory R. Madey ; John Noll, Notre Dame, United States, 2010 (Open Source Software : New Horizons), t. AICT-319, p. 308-313, DOI : 10.1007/978-3-642-13244-5_24.

BENKELTOUM (Nordine), « Open source et systèmes critiques : le cas Thales », dans *17ème Colloque de l'AIM*, dir. Association Information et Management, Bordeaux, France, 2012, s73-1, URL : <https://hal.archives-ouvertes.fr/hal-00905919> (visité le 22/09/2020).

CHARLEUX (Amel), *L'OPEN SOURCE ENTRE CONCURRENTS - Approche de la création et de l'appropriation de valeurs par les business models et la coopétition*, Theses, dir. Anne Mione, Université de Montpellier, 2019, URL : <https://hal.archives-ouvertes.fr/tel-02523313> (visité le 22/09/2020).

Liste des acronymes

ALMA**Na****CH** Automatic Language Modelling and Analysis & Computational Humanities

BnF Bibliothèque nationale de France

ENC École nationale des chartes

ENS École Normale Supérieure

INHA Institut National d'Histoire de l'Art

PSL Paris, Sciences & Lettres

UNIGE Université de Genève

UniNe Université de Neuchâtel

* * *

ALTO Analysed Layout and Text Object

API Application Programming Interface

CRF Conditional Random Fields

GROBID GeneRation Of BIbliographic Data

HTR Handwritten Text Recognition

IIIF International Image Interoperability Framework

METS Metadata Encoding and Transmission Standard

NER Named Entity Recognition

OCR Optical Character Recognition

ODD One Document Does it all

PAGE Page Analysis and Ground truth Elements

PDF Portable Document Format

SIG Système d'Information Géographique

TAL Traitement automatique des langues

TEI Text Encoding Initiative

XML eXtensible Markup Language

XSL-FO eXtensible Stylesheet Language - Formatting Objects

XSLT eXtensible Stylesheet Language Transformations

Introduction

Dans un billet de blog intitulé « Retour sur la datavisualisation “Sur la piste des ventes d’antiques” » datant du mois de mai dernier, Antoine Courtin, chef du service numérique de la recherche au département des études et de la recherche à l’INHA (Institut national d’histoire de l’art), a écrit :

le numérique doit être une opportunité de rendre le savoir plus accessible, à la fois à des publics considérés comme éloigné de la recherche engagée, mais également à l’intérieur même des communautés scientifiques. Ainsi, il faut réfléchir à ce que doit être la « narrativité scientifique numérique ».¹

À travers ces mots, il invite à questionner la manière dont les sources et le savoir sont mis en valeur par les outils numériques. Il rappelle également que la mise en ligne d’outils et de sources doit servir un but et avoir une utilité. Mis à part ces réflexions, Antoine Courtin présente dans ce billet la datavisualisation « Sur la piste des ventes d’antiques » réalisée par l’équipe du service numérique de la recherche de l’INHA dans le cadre du programme de recherche « Répertoire des ventes d’antiques en France au XIX^e siècle ». Cette datavisualisation cartographie le parcours d’objets antiques, de leur lieu de création ou de découverte jusqu’à leur lieu de conservation actuel. Elle répertorie notamment les objets vendus aux enchères au XIX^e siècle et leurs propriétaires. Ce projet, prenant la forme d’une carte associée à des données et à une ligne temporelle, n’est pas sans rappeler, de par sa forme, celui lancé quelques années auparavant par Béatrice Joyeux-Prunel, Catherine Dossin et Léa Saint-Raymond : Artl@s². Celui-ci fédère différents sous-projets dont le principal est BasArt³, une base de données géoréférencées qui recense des expositions et leurs catalogues du XIX^e siècle à nos jours. De la même façon que la datavisualisation « Sur la piste des ventes d’antiques » expose ces données, l’interface cartographique de BasArt permet de localiser des œuvres exposées, des exposant·e·s et des expositions, dont les données sont tirées de catalogues d’exposition.

Parmi le panorama des réalisations numériques liées à l’histoire de l’art, Artl@s est

1. Antoine Courtin, « Retour sur la datavisualisation “Sur la piste des ventes d’antiques” », *Numérique et recherche en histoire de l’art*, <https://numrha.hypotheses.org/743> (visité le 12/09/2020)

2. Le projet Artl@s a été mis en ligne courant 2018. Voir « BasArt : une base mondiale de catalogues d’expositions (19e-21e siècles), accessible à tou.te.s » Artl@s, <https://artlas.huma-num.fr/fr/bases-en-acces-libre/> (visité le 10/08/2020).

3. *ibid.*

unique de par sa volonté de proposer une histoire de l'art globale à travers l'étude de la mondialisation de l'art à partir des catalogues d'exposition mais aussi d'allier cette vision de la discipline aux humanités numériques⁴. Dans le domaine de l'histoire de l'art, diverses structures donnent accès à des sources de type catalogographique : la principale institution répertoriant en ligne des catalogues de vente aux enchères est le Getty Research Institute, dont le Getty Provenance Index Sale Catalogues recense les catalogues de vente entre 1650 et 1945. À côté de cette source centralisée, de nombreuses bibliothèques ont numérisé leurs collections comme la BnF (Bibliothèque nationale de France), la Royal Academy de Londres⁵, le Rijksbureau voor Kunsthistorische Documentatie, la bibliothèque numérique du Belvedere⁶, la bibliothèque de l'INHA⁷ ou encore la bibliothèque universitaire d'Heidelberg. Néanmoins, peu d'entre elles ont créé une base de données pour permettre aux chercheur·se·s d'interroger le contenu de ces sources. Par exemple, le musée d'Orsay a réalisé la base Salons⁸ qui répertorie les entrées des livrets de Salons artistiques parisiens – comme le Salon de la Société des Artistes français, celui de la Société nationale des Beaux-Arts et le Salon d'Automne –, de régions, ainsi que des principales expositions de groupes entre 1673 et 1914. Un autre projet de mise en ligne des données tirées de catalogues est celui de la publication des catalogues de vente allemands dans le Getty Provenance Index, un des premiers projets d'humanités numériques du Getty Research Institute⁹. Pour intégrer directement les données du catalogue dans la base de données, le Getty Research Institute a créé un *workflow* en cinq étapes :

1. Numérisation et OCRisation des catalogues.
2. Parsage de la transcription de l'OCR grâce à un logiciel qui place les informations dans un fichier Excel, puis normalisation des données par des algorithmes.
3. Correction des données à la main.
4. Ajout des données dans la base de données.
5. Mise en ligne des données.

L'idée de cette chaîne de traitement, créée en 2013, est aujourd'hui reprise par plusieurs équipes de recherches qui souhaitent récupérer les données de catalogues pour les injecter dans une base de données interrogeable. Même si les technologies ont évolué depuis la

4. « Base Artl@s », *BasArt*, <https://artlas.huma-num.fr/map/#/> (visité le 10/08/2020).

5. « Search for Exhibition catalogues », *Royal Academy*, https://www.royalacademy.org.uk/art-artists/search/exhibition-catalogues?all_fields=exhibition&commit=Search&date=&form=exhibition_catalogues&q=exhibition&title=&utf8=%E2%9C%93 (visité le 12/09/2020).

6. *Belvedere Digital Library*, <https://digitale-bibliothek.belvedere.at/viewer/> (visité le 12/09/2020).

7. « Collections : Imprimés », *Collections numérisées de la bibliothèque de l'INHA* , <https://bibliotheque-numerique.inha.fr/collection-imprimes> (visité le 13/09/2020).

8. « Accueil », *Base Salons*, <http://salons.musee-orsay.fr/> (visité le 10/08/2020).

9. Ruth Cuadra et Suzanne Michels, *Publishing German Sales, A Look under the Hood of the Getty Provenance Index*, The Getty Iris, 17 avr. 2013, <https://blogs.getty.edu/iris/publishing-german-sales-a-look-under-the-hood-of-the-getty-provenance-index/> (visité le 04/08/2020).

création de ce *workflow*, sa méthodologie est toujours actuelle puisqu'elle s'appuie sur la récupération de la transcription de textes, qui est ensuite soumise à des algorithmes d'intelligence artificielle pour parser l'information avant d'être intégrée dans une base de données. C'est ce que souhaite faire Béatrice Joyeux-Prunel pour automatiser l'entrée des données issues de catalogues d'exposition dans BasArt, tâche jusqu'alors réalisée à la main par les membres de l'équipe d'Artl@s. Cette automatisation permettrait de décharger les membres de l'équipe de cette tâche chronophage. Pour cela, la fondatrice d'Artl@s a fait appel à deux équipes travaillant depuis quelque temps sur l'encodage automatique de sources catalographiques : celle de *GROBID-dictionaries*, qui a développé un logiciel de *machine learning* qui encode automatiquement des dictionnaires, et celle d'editiones, qui a testé le logiciel *GROBID-dictionaries* sur des catalogues de vente de manuscrits l'an passé. Leurs travaux ont permis de montrer qu'il est possible d'encoder, grâce au *machine learning*, des sources structurées et normalisées¹⁰. Yann Sordet définit d'ailleurs le catalogue comme étant une source où les données sont organisées selon une « séquentialisation verticale »¹¹, c'est-à-dire dont les entrées sont découpées en sous-parties distinctes et organisées de manière verticale. De plus, l'information délivrée y est répétitive et structurée, deux conditions favorables à l'apprentissage machine qui s'appuie sur des *patterns* réguliers.

Le stage, dont le présent mémoire fait état, s'est déroulé au sein de l'équipe d'Artl@s, sous la direction de Béatrice Joyeux-Prunel et a été supervisé par Simon Gabay, chargé d'enseignement à l'UniNe (Université de Neuchâtel). Durant quatre mois, il s'agissait de s'appuyer sur le précédent travail de Lucie Rondeau du Noyer, anciennement stagiaire pour le projet editiones, qui s'est appliquée à tester l'outil *GROBID-dictionaries* pour encoder automatiquement des catalogues de vente de manuscrits, puis à penser une chaîne de traitement pour traiter un catalogue de sa forme numérisée jusqu'à son insertion dans une base de données¹². Il faut donc produire un *workflow* prenant en compte toutes les étapes nécessaires à l'encodage d'un catalogue d'exposition, depuis la récupération d'une version numérique du catalogue jusqu'à son encodage en XML-TEI pour ensuite le préparer à son versement dans la base de données BasArt. Par ailleurs, ce *workflow* se doit d'être le plus accessible possible puisqu'il sera par la suite réemployé par les membres de l'équipe d'Artl@s, dont certain·ne·s sont peu formé·e·s aux outils numériques. De plus,

10. Simon Gabay, Lucie Rondeau du Noyer et Mohamed Khemakhem, « Selling autograph manuscripts in 19th c. Paris : digitising the Revue des Autographes », dans *X Convegno AIUCD*, Milan, 2020, <https://hal.archives-ouvertes.fr/hal-02388407> (visité le 04/08/2020).

11. Yann Sordet, « Pour une histoire des catalogues de livres : matérialités, formes, usages » dans *De l'argile au nuage, une archéologie des catalogues : IIe millénaire av. J. C. - XXIe siècle*, avec la coll. de Bibliothèque Mazarine et Bibliothèque publique et universitaire, [Paris] Genève, 2015, p. 16.

12. Se référer au mémoire qu'elle a soutenu à l'École nationale des chartes. Lucie Rondeau du Noyer, *Encoder automatiquement des catalogues en XML-TEI. Principes, évaluation et application à la Revue des autographes de la librairie Charavay*, mémoire de master « Technologies numériques appliquées à l'histoire », dir. Thibault Clérice et Simon Gabay, Paris, École nationale des chartes, 2019.

les missions de ce stage s'insèrent dans un cadre de recherche inter-institutionnel puisque les résultats obtenus avec *GROBID-dictionaries* sur les catalogues d'exposition intéressent également les équipes travaillant autour de *GROBID-dictionaries*. Ce stage présente aussi l'occasion de tester la nouvelle version de l'outil de *machine learning* qui lit un autre format de fichiers. Il s'inscrit donc pleinement dans un contexte de recherche en laboratoire : il comporte à la fois une partie de recherche, où sont évalués les résultats obtenus, et une partie plus technique qui aboutit à la livraison d'une chaîne de traitement fonctionnelle et réutilisable.

Le présent mémoire retrace l'ensemble du *workflow* mis en place pour le projet Artl@s, tout en mettant en lumière certains aspects techniques et certaines réflexions portées sur les enjeux du *workflow*.

Le stage étant tout d'abord au cœur de problématiques inter-institutionnelles, il importe de contextualiser le cadre dans lequel il s'est déroulé. Cette première partie revient sur le projet Artl@s et les équipes avec lesquelles ses membres collaborent pour travailler sur l'encodage automatique des catalogues. Elle présente également une typologie des sources utilisées dans le cadre du stage ainsi que les différentes modélisations d'encodages proposées pour ces sources.

Ensuite, une deuxième partie s'attarde sur certains aspects techniques du *workflow* liés à l'intégration d'un nouveau type de fichiers dans *GROBID-dictionaries*. Ainsi, après avoir rappelé le fonctionnement et les enjeux de l'outil de *machine learning*, nous reviendrons sur les enjeux techniques et les résultats obtenus par certaines étapes du *workflow*.

Enfin, dans une dernière partie nous aborderons l'accessibilité du *workflow*. Après un retour en détails sur chacune de ses étapes, nous questionnerons son accessibilité avant de porter une dernière réflexion sur son utilisation et ses perspectives.

Première partie

Encoder automatiquement des catalogues : un enjeu inter-institutionnel

Chapitre 1

Équipes et projets

Encoder automatiquement des sources de type catalographique est un enjeu commun à plusieurs projets en humanités numériques. L'idée est à l'initiative de membres de l'équipe-projet ALMAnaCH (*Automatic Language Modelling and Analysis & Computational Humanities*) de l'Inria, qui ont développé l'outil de *machine learning* GROBID pour extraire automatiquement des informations provenant de références bibliographiques. Le projet s'est ensuite étendu aux dictionnaires puis aux catalogues, sources dont la structure typographique est proche de celle des dictionnaires. Deux équipes mènent actuellement des travaux sur l'encodage automatique des catalogues : celle d'e-ditiones pour les catalogues de vente de manuscrits et celle d'Artl@s – au sein de laquelle s'est déroulé ce stage – pour les catalogues d'exposition.

1.1 Le projet Artl@s

1.1.1 Un projet de plus de dix ans

Retour sur la genèse du projet et ses objectifs

À la fin des années 2000, Béatrice Joyeux-Prunel, alors maîtresse de conférences à l'ENS (École Normale Supérieure), constate un manque dans le panorama des bases de données existantes sur les sources de l'histoire de l'art : aucune d'entre elles ne répertorie de manière globale les expositions et leurs catalogues¹. Outre la base Salons du Musée d'Orsay² – créée en 2006 par Catherine Chevillot et Georges Vigne, grâce à l'aide d'un partenariat entre le musée et l'INHA – qui répertorie le contenu des livrets de Salons français entre 1673 et 1914, il existe peu de bases de données de ce type développées à grande échelle³.

1. « BasArt : une base mondiale de catalogues d'expositions (19e-21e siècles)...

2. « Accueil », *Base Salons...*

3. Le Getty Research Institute était, dans les années 2000, la seule institution à avoir créé une base de données d'envergure sur les sources primaires en histoire de l'art : le Getty Provenance Index Sale

Face à ce constat, Béatrice Joyeux-Prunel, aujourd’hui Professeure ordinaire à l’UNIGE (Université de Genève), fonde avec Catherine Dossin, *associate professor* à l’Université Purdue, et Léa Saint-Raymond, postdoctorante à l’ENS, le projet Artl@s. Il fédère différents projets de recherche et réalisations numériques sur la mondialisation artistique et culturelle – dont BasArt⁴, une base de données géoréférencées répertoriant des catalogues d’expositions du XIX^e siècle à nos jours – dans l’optique de décentraliser la recherche en histoire de l’art⁵. En proposant une base de données regroupant des sources issues du monde entier, Artl@s souhaite apporter une approche décoloniale, tout en facilitant l’accès. Les catalogues d’exposition sont en effet une source majeure pour les chercheur·se·s souhaitant travailler sur la mondialisation et la géographie de l’art, les transferts culturels et la circulation des œuvres et des artistes⁶.

Artl@s met également à disposition des chercheur·se·s et des institutions des outils et des bouts d’interface personnalisables pour qu’ils créent leur propre base de données géoréférencée en fonction des problématiques qui les intéressent. C’est ainsi que furent déployés GeoMAP⁷, le Répertoire des Pensionnaires de l’Académie de France à Rome⁸ et Picasso-Méditerranée (2017-2019)⁹.

À côté de ces réalisations numériques, Artl@s propose des ateliers de formation aux humanités numériques pour les chercheur·se·s travaillant au sein du projet et pour celles et ceux souhaitant s’y sensibiliser. L’équipe organise aussi des séminaires et des colloques autour de la mondialisation de l’art, des circulations artistiques et des humanités numériques liées à l’histoire de l’art. Parallèlement, elle publie depuis 2012 la revue *Artl@s Bulletin*¹⁰ pour exposer les résultats de recherche de ses chercheur·ses.

Nouveaux objectifs

Depuis 2017, l’équipe d’Arnl@s souhaite agrandir le projet à travers l’amélioration du volet textuel – consacré aux données entrées dans BasArt – et la création d’un volet image à travers le projet Visual Contagions¹¹.

Catalogues répertorie les catalogues de vente aux enchères entre 1650 et 1945.

4. « Base Artl@s », *BasArt...*

5. En histoire de l’art, la recherche est principalement axée sur les centres artistiques occidentaux. Depuis les années 1990, un pan de la recherche tend à se défaire de cette vision à travers une étude de l’art trans-nationale et celle des cultures non occidentales.

6. Béatrice Joyeux-Prunel, « Bases de données et gestion de projets en humanités numériques Les dessous du projet Artl@s », *Biens symboliques / Symbolic Goods*, 2018, <https://www.biens-symboliques.net/242> (visité le 30/07/2020).

7. GeoMAP répertorie les expositions parisiennes qui ont eu lieu entre 1815 et 1955. Voir « Géographie du marché de l’art parisien », *GeoMAP*, <https://paris-art-market.huma-num.fr/> (visité le 10/08/2020).

8. *ACAD de France à Rome*, <https://acad-artlas.huma-num.fr/> (visité le 10/08/2020).

9. « Accueil », *Picasso-Méditerranée*, <https://picasso-mediterranee.org/index.html#fr/map/> (visité le 10/08/2020).

10. « Journal Home », *Arnl@s Bulletin*, <https://docs.lib.purdue.edu/artlas/> (visité le 10/08/2020).

11. « Home », *Imago/Visual Contagions*, <https://www.imago.ens.fr/> (visité le 10/08/2020).

Jusqu’alors les données entrées dans BasArt étaient entrées manuellement par des chercheur·se·s qui dépouillaient des catalogues d’exposition pour renseigner les informations concernant l’exposition elle-même et, le cas échéant, la liste des exposant·e·s et des œuvres exposées. Ce type de collecte étant chronophage et producteur d’erreurs, l’équipe a décidé de se tourner vers le *machine learning* pour automatiser le traitement des catalogues et l’entrée des données dans la base. Pour cela, la solution envisagée est d’utiliser la TEI (*Text Encoding Initiative*)¹² comme format pivot pour passer du catalogue numérisé aux données injectées dans BasArt. Le catalogue sera automatiquement encodé en XML-TEI grâce à l’outil *GROBID-dictionaries* afin d’en extraire l’information pour ensuite l’injecter dans la base de données. Ce travail, qui est l’objet du présent stage, va permettre de faciliter la récupération des entrées des catalogues et d’augmenter le nombre de données insérées dans BasArt.

D’autre part, l’avancée des *computer vision*¹³ a amené l’équipe d’Artl@s à s’intéresser aux « contagions visuelles », c’est-à-dire à la mondialisation visuelle ou encore à la manière dont les images circulent. Le projet Visual Contagions propose d’étudier ces circulations visuelles grâce à l’intelligence artificielle. Les résultats de ces expérimentations permettraient d’identifier les images influentes et leurs voix de circulations afin d’en comprendre les processus¹⁴. Ce volet s’appuie sur un corpus d’images et d’illustrations qui seront extraites de catalogues d’exposition et de périodiques puis taguées de mots-clés identifiant les motifs de l’image par des algorithmes. L’objectif sera ensuite de tester l’algorithme d’EnHerit (*Enhancing Heritage Image Databases*)¹⁵ développé par Mathieu Aubry, chercheur à l’École des Ponts Paris Tech, sur ce corpus afin de regrouper les images similaires pour les étudier.

1.1.2 La base de données BasArt

La base de données BasArt, financée par l’ANR (Agence nationale pour la Recherche) Jeunes Chercheurs - Jeunes Chercheuses, l’ENS, l’université PSL (Paris, Sciences & Lettres) et le labex TransferS¹⁶, a été créée en 2011, mise en ligne en 2016 et accessible publiquement en 2018. Elle regroupe et centralise les données de catalogues d’exposi-

12. La TEI définit des recommandations pour encoder des documents textuels en utilisant le langage XML (*eXtensible Markup Language*). D’après Lou Burnard, ce langage permet de représenter simplement des données structurées et de marquer les parties textuelles spécifiques grâce à des balises. Voir Lou Burnard, *Qu'est-ce que la Text Encoding Initiative ?*, Marseille, OpenEdition Press, 2015, p. 13.

13. La *computer vision*, ou vision par ordinateur en français, désigne les différentes techniques permettant aux ordinateurs de voir et de comprendre le contenu d’images. Il s’agit d’une sous-catégorie du *machine learning*.

14. Béatrice Joyeux-Prunel, « Visual Contagions, the Art Historian, and the Digital Strategies to Work on Them », *Artl@s Bulletin*, 8-3, 2019, p. 133.

15. « Présentation », *EnHerit*, <https://enherit.enpc.fr/> (visité le 10/08/2020).

16. Le labex TransferS est un laboratoire d’excellence qui rassemble des équipes de recherche de l’ENS et du Collège de France autour de la question des transferts culturels. Voir « Présentation », *Labex TransferS*, <http://www.transfers.ens.fr/-presentation-> (visité le 11/08/2020).

tion issus du monde entier, telles celles des biennales de Sao Paulo, de Johannesburg et d'Alexandrie.

Implémentation

BasArt est une base de données PostGIS, c'est-à-dire une base de données PostgreSQL avec un SIG (Système d'Information Géographique). Ce type de base de données permet de traiter des informations spatiales sous forme de points, lignes ou polygones et de générer des cartes. Le modèle conceptuel de BasArt a été réalisé par Yann Le Boulangier, maître de conférences en Sciences de l'informatique à l'IUT d'Avray. Ce modèle s'avère être complexe à cause de la multiplicité des tables dues aux nombreuses informations contenues dans un catalogue d'exposition et au géoréférencement des adresses. Ce dernier entraîne la création d'un emboîtement de tables allant du lieu au pays, chaque lieu étant lié à une ville, elle-même liée à un état ou une région qui est enfin rattaché·e à un pays. D'autre part, la diversité des informations contenues dans un catalogue d'exposition a amené à la création de nombreuses tables couvrant tous les champs pouvant être rencontrés dans ce type de sources. Certaines tables, spécifiques à un certain type de catalogue, se retrouvent régulièrement vides¹⁷, quand d'autres informations, par souci de respect de la source, sont redondantes¹⁸.

De plus, BasArt repose sur un système *loosely coupled*¹⁹, c'est-à-dire un système dont les composants sont peu liés les uns aux autres. Cela permet de remplacer un composant par un autre donnant le même type de service. Ce type d'implémentation est moins contraignant vis-à-vis de la plateforme, du langage, du système d'exploitation ou encore de l'environnement de construction du logiciel. Néanmoins, il peut engendrer des problèmes de cohérence des données et impose des protocoles de coordination supplémentaires si des changements sont opérés. Cette flexibilité offre la possibilité à l'équipe d'Artl@s de pouvoir opérer des changements au sein de la structure sans toutefois devoir tout recommencer. Elle s'est également tournée vers des logiciels libres, sans abonnement ou licence dans l'optique de pérenniser le projet.

Enfin, l'insertion de données géographiques a permis de réaliser une interface cartographique et statistique, codée en PHP et en Javascript, accessible aux historien·ne·s.

17. Par exemple, une table a été créée pour les groupes d'expositions, or même si la plupart des expositions sont des expositions de groupe, dans le sens où elles ne sont pas des expositions individuelles, elles ne sont pas toutes affiliées à un groupe artistique.

18. Certaines informations géographiques sont doublement renseignées : une première case permet d'indiquer l'adresse telle qu'elle est renseignée dans le catalogue, tandis qu'une autre précise les données de géolocalisation de l'adresse.

19. Sorin Matei, « ARTL@S and BasArt : A Loose Coupling Strategy for Digital Humanities », *Artl@s Bulletin*, 1–1, 2012, p. 28.

Fonctionnalités

L'interface publique de BasArt prend la forme d'une base de données interrogeable grâce à une barre de recherche simple ou avancée. Il est possible de faire des recherches sur les expositions, les exposant·e·s et les œuvres. Les résultats de la recherche apparaissent sous la forme d'une liste accompagnée d'une carte et d'un graphique. L'utilisateur·rice peut contraindre la recherche à une période donnée et ainsi obtenir des résultats plus spécifiques. L'enjeu de BasArt est d'encourager les chercheur·se·s à faire des recherches sur l'histoire mondiale et sociale de l'art, tout en racontant l'histoire des œuvres et des artistes à travers l'utilisation de cartes²⁰. Il leur est également possible d'exporter le résultat de leur recherche sous la forme d'un fichier CSV.

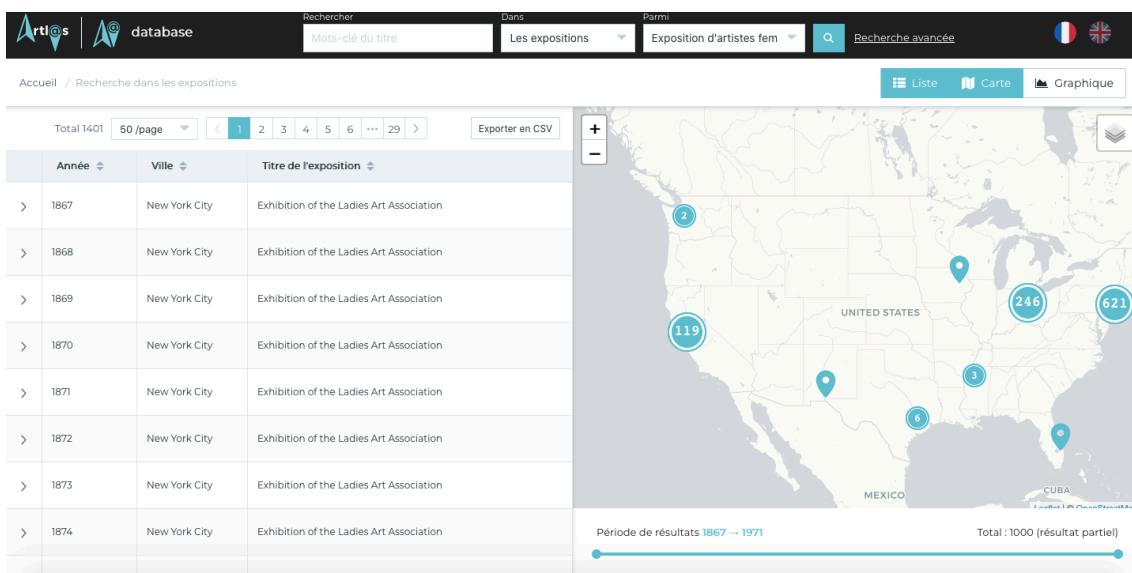


FIGURE 1.1 – Capture d'écran d'un résultat de recherche sur BasArt, août 2020.

La base de données possède aussi un accès contributeur·rice basé sur le bénévolat. Ce sont principalement des chercheur·se·s spécialisé·e·s en histoire de l'art qui y insèrent le fruit de leurs recherches. La base est également alimentée grâce à des travaux réalisés en collaboration avec des institutions patrimoniales et des projets de recherche. Avant d'être mises en ligne, les données sont vérifiées par l'équipe afin d'en contrôler l'aspect scientifique et le respect de la source.

En une dizaine d'années, Artl@s a su se faire une place dans le panorama des humanités numériques, et plus particulièrement dans celui des projets numériques en histoire de l'art. Après avoir déployé une base de données – consultée par des historien·ne·s de l'art, des acteur·rice·s du marché de l'art et des institutions patrimoniales²¹ – l'enjeu est

20. Cette logique de faire parler les cartes est retrouvée dans le projet Sur la piste des œuvres antiques réalisé par l'INHA. Voir *Sur la piste des œuvres antiques*, <https://ventesdantiques.inha.fr/> (visité le 11/08/2020).

21. B. Joyeux-Prunel, « Bases de données et gestion de projets en humanités numériques...

désormais d'augmenter la production des données et de décharger les collaborateur·rice·s à travers la création d'un *workflow* pour encoder automatiquement les catalogues d'exposition en XML-TEI, avant de les verser dans BasArt. Ce *workflow* doit être le plus accessible possible pour permettre aux chercheur·se·s, dont la formation au numérique est réduite, de l'utiliser. Enfin, ce travail s'insère dans un projet plus général sur l'encodage automatique des catalogues, sur lequel travaillent également des membres de l'équipe-projet ALMAAnaCH et du projet e-ditiones.

1.2 Autres équipes travaillant sur l'encodage automatique des catalogues

1.2.1 L'équipe-projet ALMAAnaCH

ALMAAnaCH est une équipe-projet de l'Inria, l'Institut national de recherche en sciences et technologies du numérique, travaillant sur le traitement automatique des langues et les humanités computationnelles²². Les membres d'ALMAAnaCH sont amené·e·s à développer des logiciels et des ressources de TAL (Traitement automatique des langues) utiles à la recherche et à l'analyse linguistique. Certain·e·s d'entre elles ou eux travaillent sur l'extraction automatique d'informations linguistiques et lexicales. C'est le cas de Mohamed Khemakhem, doctorant à l'université Paris Diderot - Paris 7 et doctorant associé à l'Inria et au Centre March Bloch, qui a développé l'outil *GROBID-dictionaries*.

Présentation des membres de l'équipe travaillant sur *GROBID-dictionaries*

Deux membres de l'équipe-projet ALMAAnaCH travaillent sur *GROBID-dictionaries* : Mohamed Khemakhem et Laurent Romary. Ce dernier est un membre permanent de l'Inria dont les recherches sont axées sur les modèles de données linguistiques, les ressources lexicales, l'extraction d'information et la *data mining*. Il a précédemment travaillé avec Luca Foppiano, ingénieur au *National Institute for Materials Science*, et Patrice Lopez, fondateur de science-miner²³, sur GROBID, un outil de *machine learning* permettant d'extraire des informations de documents structurés et de les encoder automatiquement au format XML-TEI²⁴. Cet outil s'appuie sur un modèle CRF (*Conditional Random Fields*) en cascade, c'est-à-dire qu'il extrait et encode les informations en allant du plus général au plus petit²⁵. Grâce à cela, ils ont pu extraire des informations issues des métadonnées

22. « Présentation », *ALMAAnaCH*, <https://team.inria.fr/almanach/fr/> (visité le 12/08/2020).

23. science-miner propose des outils *open source* et des services en recherche et développement autour du *machine learning*. Voir « Home », *science-miner*, <http://science-miner.com/> (visité le 12/08/2020).

24. Laurent Romary et Patrice Lopez, « GROBID - Information Extraction from Scientific Publications », *ERCIM News*, 100, 2015, p. 2.

25. GROBID encode premièrement le corps de texte dans `<body>` puis il opère un encodage de plus en plus fin.

de sources bibliographiques et des références bibliographiques.

Suite aux résultats concluants de GROBID, Laurent Romary a souhaité étendre la recherche sur l'encodage automatique des documents structurés à d'autres sources, tels les dictionnaires. Il a ainsi créé avec Mohamed Khemakhem *GROBID-dictionaries*, un sous-projet de GROBID, qui permet d'extraire les informations provenant d'entrées de dictionnaires.

Objectifs

Le développement de *GROBID-dictionaries*²⁶ sous-tend différents enjeux : l'adaptation de GROBID aux sources encyclopédiques à travers la création d'un modèle XML-TEI conforme, l'annotation des données d'entraînement et la sélection des *features*²⁷ sur lesquels le modèle va s'appuyer²⁸.

Les résultats présentés à la conférence eLex 2017²⁹ sont plus ou moins bons selon les niveaux d'encodage et les sources utilisées. Ils ont permis à l'équipe de revoir la sélection des *features* et d'augmenter le nombre de données d'entraînement. L'année suivante, ils ont testé *GROBID-dictionaries* sur d'autres sources comme le *Petit Larousse Illustré*³⁰ et des entrées d'annuaires et d'almanachs³¹. Outre certains ajustements à faire en fonction des sources – comme ajouter la balise <num> pour encoder correctement les entrées des annuaires – ces exemples démontrent la flexibilité de *GROBID-dictionaries*. Son adaptabilité aux sources structurées par des entrées telles que les dictionnaires, les encyclopédies et les annuaires a mené Simon Gabay, collaborateur scientifique à l'UniNe, à s'y intéresser pour encoder des catalogues de vente de manuscrits.

1.2.2 Le projet e-ditiones

Brève présentation du projet

Le projet e-ditiones³², dirigé par Simon Gabay, recense les manuscrits français du XVII^e siècle. Il s'appuie sur deux volets : le premier porte sur les manuscrits même, tandis que le second se base sur les catalogues de vente de manuscrits. Cette source

26. Le fonctionnement de *GROBID-dictionaries* est développé plus en détails au chapitre 3.

27. Ici, les *features* correspondent à la représentation numérique d'un aspect de la donnée brute.

28. Mohamed Khemakhem, Luca Foppiano et Laurent Romary, « Automatic Ex-traction of TEI Structures in Digitized Lexical Resources using Conditional RandomFields », dans *lectronic lexicography*, eLex 2017, Leiden, 2017, p. 6-9.

29. *ibid.*

30. Hervé Bohbot, Francesca Frontini, Giancarlo Luxardo, Mohamed Khemakhem et Laurent Romary, « Presenting the Néufar Project : a Diachronic Digital Edition of the Petit Larousse Illustré », dans *GLOBALEX 2018 - Globalex workshop at LREC 2018*, Miyazaki, 2018, p. 1-6.

31. Mohamed Khemakhem, Carmen Brando, Laurent Romary, Frédérique Mélanie-Becquet et Jean-Luc Pinol, « Fueling Time Machine : Information Extraction from Retro-Digitised Address Directories », dans *JADH2018 “Leveraging Open Data”*, Tokyo, 2018.

32. « Accueil », *e-ditiones*, <https://editiones.hypotheses.org/> (visité le 13/08/2020).

secondaire s'avère être précieuse pour retrouver des manuscrits aujourd'hui perdus ou pour retracer leur histoire. Afin de pouvoir interroger facilement ces sources, Simon Gabay s'est tourné vers l'encodage automatique en XML-TEI. En 2019, il a travaillé – en collaboration avec Mohamed Khemakhem et Laurent Romary – avec Lucie Rondeau du Noyer, alors étudiante à l'ENC (École nationale des chartes), sur l'entraînement de modèles pour *GROBID-dictionaries* créés à partir de catalogues de vente de manuscrits. L'enjeu était de tester la réaction de *GROBID-dictionaries* avec ce type de sources et de voir s'il était possible de généraliser le modèle d'apprentissage. Cette recherche prend place au sein d'un travail plus large mené par Lucie Rondeau du Noyer. Elle s'est chargée de proposer une chaîne de traitement pour traiter des documents numérisés et passer du PDF au fichier XML-TEI. Son *workflow* prend en compte différentes étapes : l'OCRisation du catalogue, l'encodage automatique du document en XML-TEI grâce à *GROBID-dictionaries* ainsi que l'adaptation et l'amélioration de l'encodage en sortie de *GROBID-dictionaries*³³ pour qu'il soit conforme à celui voulu par le projet editiones³⁴.

Conclusions sur l'utilisation de *GROBID-dictionaries*

Suite aux recherches qu'elle a menées, Lucie Rondeau du Noyer est arrivée aux conclusions suivantes³⁵ :

- *GROBID-dictionaries* nécessite un certain nombre de données d'entraînement pour pouvoir encoder correctement des documents.
- Il s'adapte bien à tout document dont l'information est structurée ; il est ainsi possible de traiter toute forme de catalogue avec cet outil.
- Il n'est pas intéressant de tenter de créer un modèle général car *GROBID-dictionaries* s'appuie sur la mise en page des documents, notamment sur les séparateurs typographiques.

D'autre part, étant donné que *GROBID-dictionaries* se base sur des *features*, dont la ponctuation, il ne peut encoder des documents que jusqu'à un certain niveau (il lui est par exemple impossible de distinguer un prix d'un format, tous deux étant généralement présentés sous la forme de chiffres suivis de lettres). Néanmoins, pour les besoins du projet editiones, il est intéressant d'encoder certaines informations comme le prix de vente, la date du manuscrit vendu ou encore son format. Pour affiner l'encodage obtenu grâce au *workflow* de Lucie Rondeau du Noyer, Simon Gabay a travaillé avec Matthias Gille Levenson, doctorant à l'ENS de Lyon, sur un script python qui récupère certaines

33. *GROBID-dictionaries* étant développé pour les dictionnaires, il faut réadapter l'encodage, grâce à une feuille de transformation XSL (*eXtensible Stylesheet Language*), pour qu'il convienne à la structure des catalogues.

34. L. Rondeau du Noyer, *Encoder automatiquement des catalogues en XML-TEI...*, p. 26.

35. *ibid.*, p.64.

informations, les uniformise et les encode³⁶.

Les travaux menés conjointement par Mohamed Khemakhem, Laurent Romary, Lucie Rondeau du Noyer et Simon Gabay sur *GROBID-dictionaries* ont permis de tester son efficacité sur différents types de sources mais aussi de chercher ce qui peut en améliorer la performance. À travers l'observation des sources, ils ont conclu qu'il serait intéressant d'ajouter l'information typographique (gras, italique et taille des caractères) dans les *features* pour donner des informations supplémentaires à *GROBID-dictionaries* sur lesquelles s'appuyer.

Par ailleurs, l'emploi de l'outil pour des sources catalographiques les a poussé à repenser *GROBID-dictionaries* afin de l'adapter aux catalogues. Pour cela, ils aimeraient lancer *GROBID-cat*³⁷, un *fork* de *GROBID-dictionaries* dans lequel les balises propres à l'encodage des dictionnaires seraient remplacées par des balises adéquates à celui des catalogues.

L'objet de ce stage se retrouve au cœur de toutes ces problématiques et sert autant à l'équipe d'Artl@s, qui a besoin d'automatiser la collecte des données pour BasArt, qu'aux équipes travaillant déjà avec *GROBID-dictionaries*. L'enjeu est double : il faut à la fois adapter le *workflow* de Lucie Rondeau du Noyer au cas des catalogues d'exposition et produire et évaluer de nouvelles données prenant en compte l'information typographique pour observer le comportement de *GROBID-dictionaries* avec ces nouvelles *features*.

36. Simon Gabay et Mathias Gilles Levenson, *katabase/reconciliation*, GitHub, <https://github.com/katabase/reconciliation> (visité le 13/08/2020).

37. Mohamed Khemakhem, *grobid-cat*, GitHub, <https://github.com/MedKhem/grobid-cat> (visité le 13/08/2020).

Chapitre 2

Les sources : typologie et encodage

Les catalogues d'exposition constituent une source majeure pour l'histoire de l'art : garants de la mémoire d'une exposition¹, ils délivrent des informations sur l'événement, les œuvres et les artistes. Ils sont principalement consultés par les historien·ne·s pour des recherches monographiques mais ils permettent également d'étudier la circulation des œuvres et des images². Crés pour les publics, ils prennent la forme d'une liste structurée où l'information mentionnant les œuvres exposées y est régulière. Au fil du temps, la forme du catalogue évolue et passe de la liste sommaire au catalogue fourni : c'est à la fin du XIX^e siècle que s'esquisse la forme des catalogues d'exposition tels que nous les connaissons avec une page de titre, la liste des prêteurs, une introduction et la liste des œuvres parfois accompagnée de descriptions et d'illustrations³. À partir du milieu du XX^e siècle, sa forme évolue : à la liste d'œuvres et aux reproductions s'ajoutent des essais qui apportent d'autres points de vue à l'exposition et la contextualisent. Son nombre de pages augmente considérablement⁴ pour laisser place à une liste d'œuvres grandissante, aux nombreuses illustrations et aux textes.

Malgré la forme changeante du catalogue d'exposition, il est possible d'observer une constante dans la liste des œuvres exposées : l'information – même si elle diffère légèrement d'un catalogue à un autre – y est ordonnée. Elle se prête bien à la structuration de texte telle que le permet le langage XML-TEI.

1. Camille Morineau, « Un corpus emblématique ? Les catalogues d'exposition du musée national d'art moderne de 1947 à 1977 », dans *Cahiers du Musée national d'art moderne*, n°56/57, Paris, 1996, p. 155.

2. Béatrice Joyeux-Prunel et Olivier Marcel, « Exhibition Catalogues in the Globalization of Art. A Source for Social and Spatial Art History », dans *Artl@s Bulletin*, 4-2, 2015, p. 81.

3. Guilhem Scherf, « Le catalogue d'exposition : plaidoyer pour un genre littéraire », dans *Le catalogue dans tous ses états : actes du colloque [Rencontres de l'École du Louvre]*, 12, 13 et 14 décembre 2012, Paris, 2015, p. 243-244.

4. Il augmente d'environ 300% entre les années 1950 et 1980. Voir C. Morineau, « Un corpus emblématique ?..., p. 158.

2.1 Typologie des catalogues d'exposition

Pour les besoins du stage, il était primordial d'établir une typologie des catalogues d'exposition dans le but d'entraîner des modèles avec *GROBID-dictionaries*. Elle s'appuie sur celle dressée par Barbara Topalov – étudiante du deuxième cycle de l'École du Louvre, spécialité Documentation et humanités numériques, et stagiaire au sein du projet Artl@s – à partir de la liste qu'elle a établie recensant des catalogues accessibles en ligne⁵. Elle a distingué les expositions monographiques des expositions de groupes, elles-mêmes départagées en fonction du type des entrées, par nom d'artiste ou par titre d'œuvre.

2.1.1 Les expositions monographiques

Les catalogues d'exposition monographique sont ordonnés par une liste d'œuvres. Les entrées, parfois regroupées par technique ou sujet pictural, sont caractérisées par un numéro suivi du titre de l'œuvre. Dans certains catalogues, les œuvres font l'objet d'une description plus ou moins détaillée, généralement identifiable par des parenthèses, un retour à la ligne ou une police plus petite que celle du titre. Parmi les catalogues d'exposition individuelle recensés par Barbara Topalov, il est possible de distinguer deux types d'entrées :

- Les entrées dont le numéro et le titre sont séparés par une espace et dont les informations propres à l'œuvre sont indiquées entre parenthèses ou après un retour à la ligne. La figure A.1, extraite du *Catalogue de l'exposition des œuvres de Claude Monet* (1883), illustre bien ce type d'entrée simple où seuls les espaces, les retours à la ligne et la position du début de la ligne permettent d'identifier clairement les informations.
- Les entrées dont le numéro et le titre de l'œuvre sont séparés par un cadratin et dont les informations concernant l'œuvre sont indiquées entre parenthèses ou après un retour à la ligne. Contrairement à la figure A.1, la figure A.2, extraite du catalogue *Exposition des œuvres de Gustave Courbet à l'école des Beaux-Arts* (1882), offre une plus grande diversité d'informations typographiques : le numéro de l'entrée est séparé par un point suivi d'un cadratin tandis que chaque information concernant l'œuvre est signalée par un retour à la ligne et une police plus petite ; les dimensions du tableau étant elles-mêmes mises en valeur par un alinéa plus important.

Ces catalogues, moins fréquents que ceux des expositions de groupe, se multiplient au XX^e siècle grâce aux rétrospectives et aux expositions individuelles chez des galeristes. Ils ne sont toutefois pas représentatifs du corpus d'Artl@s, plus orienté vers les expositions de groupes, notamment internationales.

5. Cette liste prend la forme d'un tableau Excel recensant plus de 600 sources. Il a uniquement été diffusé entre les membres du projet Artl@s.

2.1.2 Les expositions de groupe

Les catalogues des expositions de groupe sont ordonnés soit par nom d'artiste, soit par titre d'œuvre. Le premier cas est plus fréquent : les artistes y sont classés par ordre alphabétique et parfois regroupés par sections organisées par salles ou techniques artistiques ; tandis que les catalogues d'exposition ordonnés par titre d'œuvre sont généralement classés au sein de sections révélatrices de l'organisation de l'exposition.

Les expositions de groupe par nom d'artiste

Le corpus d'Artl@s est majoritairement constitué de catalogues organisés par nom d'artiste. Ils sont représentatifs des diverses formes que peuvent prendre les entrées de catalogue, rendant alors difficile la généralisation d'un modèle pour ce type particulier de catalogues. Nous verrons ici quelques types d'entrées illustrant leur hétérogénéité typographique.

Les catalogues de biennale ont la particularité d'être organisés en sections par pays, elles-mêmes divisées en sous-sections liées aux techniques artistiques. Chaque nouvelle salle est généralement introduite par une présentation des artistes nationaux. Malgré des différences éditoriales observées d'un catalogue de biennale à un autre (notamment dans la langue employée et la mise en page), il est possible de remarquer une constance typographique au niveau des entrées : les noms des exposant·e·s et des numéros précédents les œuvres sont tous en gras. De même, le retour à la ligne est employé uniquement pour indiquer une nouvelle œuvre ou un nouvel artiste ; les numéros sont suivis d'une espace et les informations propres à l'œuvre sont séparées du titre grâce à l'emploi de parenthèses, d'une virgule ou d'un cadratin. Les figures A.3, extraite du catalogue *II bienal de São Paulo* (1953), et A.4, extraite du catalogue *VI. Esposizione Internazionale d'Arte DellaCittà Di Venezia* (1905), illustrent bien l'emploi systématique du gras pour le nom de l'exposant et le numéro des œuvres.

Les catalogues de Salon et de musées ont une structure moins complexe que celle des catalogues de biennales. Ils sont parfois divisés en sections par technique ou salle, mais ces dernières ne sont pas à nouveau divisées en sous-sections. Au premier abord, ces catalogues semblent être similaires : le nom de l'artiste est en majuscule, les informations biographiques sont indiquées soit sur la même ligne que le nom, soit après un retour à la ligne et la liste des œuvres exposées apparaît ensuite. Toutefois, de légères variations typographiques nous ont poussé·e·s à différencier trois types de catalogues⁶ :

- Les catalogues sans gras ni italique : les différentes informations des entrées de ces catalogues sont repérables grâce aux retours à la ligne et à la position de chaque début de ligne. Par exemple, la figure A.6, extraite du *Catalogue of the inaugural*

6. Cette typologie n'est pas exhaustive, il est possible de dresser d'autres catégories de catalogues en fonction de la typographie et de la ponctuation.

exhibition du musée de Tolède (États-Unis), montre une entrée débutant par le nom de l'artiste en majuscules, puis une courte biographie, le numéro de l'œuvre, son titre et sa provenance ; chaque information étant marquée par un saut de ligne.

- Les catalogues avec biographie en italique : les entrées débutent par le nom de l'artiste en majuscules suivi du prénom, viennent ensuite les informations biographiques en italique, parfois marquées par un retour à la ligne. Les œuvres sont indiquées par un numéro suivi d'un point, du titre et parfois d'informations complémentaires. Les figures A.7, extraite du *Catalogue de l'exposition des œuvres d'art d'artistes vivants : Strasbourg (1884)*, et A.8, extraite du *Catalogue des peintures, miniatures, aquarelles, dessins, sculptures et lithographies exposés à Nancy (1847)*, illustrent bien ce type d'entrée fréquente dans les catalogues du XIX^e siècle.
- Les catalogues avec numéro en gras : La série des catalogues d'exposition de la Société des artistes indépendants est représentative de ce type d'entrées⁷. Après le nom de l'artiste en majuscules vient le prénom entre parenthèses. Les informations biographiques suivent le nom sur la même ligne et les œuvres sont listées en-dessous, bien identifiables grâce à l'emploi du gras sur les numéros.

Les expositions de groupe par titre d'œuvres

Outre les catalogues ordonnés par nom d'artiste, il existe des catalogues organisés par titre d'œuvre. Ce type d'entrée, moins fréquent, est retrouvé dans les catalogues de la *Royal Academy* de Londres, au Canada ainsi que dans certains catalogues d'arts décoratifs. Cette disposition des entrées se rapproche de celle des catalogues de vente de livres anciens et de manuscrits⁸. Elles débutent par un numéro suivi du titre de l'œuvre puis du nom de l'artiste, aligné à droite et en italique pour les catalogues de la *Royal Academy*⁹ ou après un retour à la ligne dans le cas du *Catalogue illustré de la section japonaise à l'exposition internationale des arts décoratifs et industriels modernes (1925)*.

La diversité des catalogues d'exposition nous a mené·e·s à les catégoriser en fonction de l'organisation des informations, de la mise en page des entrées et de la typographie. Comme *GROBID-dictionaries* s'appuie sur ce type d'informations, il a fallu affiner la typologie dressée par Barbara Topalov et regrouper des catalogues présentant des signes typographiques similaires. La typologie dressée précédemment n'est d'ailleurs pas exhaustive puisqu'elle a été établie à partir d'un échantillon restreint de catalogues d'exposition. Il est possible de proposer de nouveaux types, de même que d'en regrouper en fonction des résultats obtenus avec le logiciel de *machine learning*.

7. Voir l'exemple de la figure A.9.

8. Barbara Topalov, Simon Gabay, Béatrice Joyeux-Prunel, Laurent Romary et Lucie Rondeau du Noyer, *Automating Artl@s - extracting data from exhibition catalogues*, 2020, p.4.

9. Voir l'exemple de la figure A.10.

2.2 Du catalogue au catalogue d'exposition : modélisation de l'encodage en XML-TEI

Les fichiers produits par *GROBID-dictionaries* étant encodés en XML-TEI, il était important de penser l'encodage des catalogues d'expositions, à la fois pour les besoins du projet Artl@s – qui utilise la TEI comme format pivot avant de pouvoir injecter les données dans BasArt – et pour ceux de *GROBID-dictionaries*. De même, dans la volonté de fédérer les recherches autour de l'encodage automatique des catalogues, Laurent Romary souhaite proposer un modèle commun qui permettrait d'encoder différents types de catalogues.

Comme les recommandations P5 du Consortium TEI¹⁰ ne proposent pas de modèle précis pour encoder des catalogues, il a fallu chercher les balises les plus pertinentes pour retranscrire la structure des entrées et leurs informations, tout en s'appuyant sur le travail de Lucie Rondeau du Noyer¹¹. Parmi les objets encodables en XML-TEI, le catalogue se rapproche plus de la liste, même si Brent Nelson rappelle qu'une entrée n'est pas tout à fait l'équivalent d'un item de liste¹². Son point de vue rejoint ainsi celui de Laurent Romary et justifie sa volonté de créer un encodage approprié à ce type de documents.

2.2.1 Encoder des catalogues d'exposition

La typologie dressée précédemment a permis de montrer les diverses formes et informations d'une entrée de catalogue d'exposition. Malgré leur hétérogénéité, les caractéristiques de ces entrées se recoupent entre elles et sont plus ou moins récurrentes d'un catalogue à un autre. De plus, leur structure reste globalement identique. La liste des entrées de catalogues est donc encodée dans une `<list>`, au sein de laquelle se trouvent des `<entry>`¹³. Pour pallier au problème des sections (par salle ou par technique), il a été décidé de les encoder dans la balise `<head>`, afin de les indiquer en tant que nom de liste¹⁴.

Mis à part les entrées des catalogues d'exposition individuelle¹⁵, toutes sont composées de deux parties distinctes relevant de l'exposant et de l'œuvre. Une `<entry>` – numérotée grâce à l'attribut `@n` – est donc composée d'un `<desc>`, où sont indiquées les informations biographiques de l'exposant, et d'un ou plusieurs `<item>` dans lesquels se

10. TEI Consortium, *TEI P5 : Guidelines for Electronic Text Encoding and Interchange*, 2020, <https://zenodo.org/record/3413524> (visité le 30/07/2020).

11. L. Rondeau du Noyer, *Encoder automatiquement des catalogues en XML-TEI...*, p. 49-51.

12. Brent Nelson, « Curating Object-Oriented Collections Using the TEI », dans *Journal of the Text Encoding Initiative-9*, 2016, <http://journals.openedition.org/jtei/1680> (visité le 04/08/2020).

13. Les réflexions sur le présent encodage concernent uniquement le `<body>`, le `<header>` des catalogues d'exposition pour le projet Artl@s a été pensé par Auriane Quoix, étudiante du master TNAH et également stagiaire au sein du projet.

14. Nous avions également pensé à la balise `<superEntry>` qui permet de regrouper des `<entry>` entre elles, mais ce type de balise est plus adaptée aux dictionnaires qu'aux catalogues.

15. Dans le cas d'une exposition monographique, les entrées sont uniquement composées d'`<item>`.

trouvent le titre de l'œuvre et un éventuel cartel. Ensuite, le <desc> est divisé en deux parties distinctes :

- La balise <name> permet d'indiquer le nom de l'artiste et de différencier le nom du prénom grâce aux balises <surname> et <forename>. Elle est complétée par un attribut @role dans lequel est précisé le rôle de l'artiste, ici comme exposant.
- La balise <trait> suivie de <p> regroupe toutes les informations biographiques concernant l'exposant. En fonction de celles qui sont renseignées et récupérées dans BasArt, il est possible d'ajouter les balises suivantes : <birth>, <death>, <residence>, <education>, <affiliation> (permet de préciser si l'artiste a fait partie d'un groupe) et <nationality>.

Enfin, l'<item>, également numéroté avec un attribut @n, peut être divisé en trois parties selon les informations données :

- La balise <num> permet d'indiquer le numéro de l'<item>.
- La balise <title> renseigne le titre de l'œuvre¹⁶.
- La balise <desc> concerne la partie descriptive de l'œuvre. Il est possible d'en-coder certaines informations récupérées dans BasArt avec les balises suivantes : <material> (technique artistique), <dimensions>, <date>, <history> (pour préci-ser sa provenance), <measure> (prix) et <note> (pour indiquer les éventuelles notes concernant l'œuvre).

Cet encodage, dont un exemple est présenté ci-dessous, a été pensé pour transcrire au mieux la structure des catalogues d'exposition tout en prenant en compte les informations récupérées dans la base de données d'Artl@s.

```

<entry n="11">
  <desc>
    <name role="exhibitor">
      <surname>BARBIER</surname>,
      <forename>Ernest</forename>
    </name>
    <trait>
      <p><birth>né à Chartres</birth>. -<residence>131, rue de
      Fontenay. Vincennes</residence></p>
    </trait>
  </desc>
  <item n="73">
    <num>73</num>

```

16. Ces deux première balises sont toujours remplies, sauf exception.

```

<title>Fleurs et Fruits</title>
</item>
<item n="74">
    <num>74</num>
    <title>Fleurs</title>
</item>
</entry>
```

FIGURE 2.1 – Exemple d'une entrée encodée extraite du catalogue de la *Société des artistes indépendants. 8, Catalogue des œuvres exposées*, Paris, 1892.

Pour permettre la validation de ce type d'encodage, il a fallu créer une ODD (*One Document Does it all*)¹⁷¹⁸ validant des enchaînements de balises jusqu'alors impossibles par les recommandations P5 de la TEI. Par exemple, une `<list>` est composée d'`<item>` et non d'`<entry>`, nous avons donc forcé cet enchaînement en intégrant la balise `<entry>` comme sous-élément `<list>` avec la spécification de schéma suivante :

```

<elementSpec ident="list" mode="change">
    <content>
        <alternate minOccurs="0" maxOccurs="unbounded">
            <textNode/>
            <elementRef key="head"/>
            <elementRef key="entry"/>
        </alternate>
    </content>
</elementSpec>
```

Il est indiqué dans `<content>` le « contenu légal de l'élément »¹⁹, en précisant ici qu'une `<list>` peut contenir du texte (`<textNode/>`), un titre (`<elementRef key="head"/>`) et des entrées (`<elementRef key="entry"/>`). D'autres spécifications de schéma de ce genre ont été ajoutées à l'ODD pour obtenir un document valide, conforme à nos attentes.

Ce modèle d'encodage sert de base pour modifier celui qui est récupéré en sortie de *GROBID-dictionaries*²⁰ et pour la création d'un modèle général pour encoder les catalogues.

17. L'ODD est un langage définissant et documentant les règles d'un projet XML-TEI. Il est possible de personnaliser l'ODD pour qu'elle soit propre à un projet particulier.

18. Cette ODD a été réalisée avec Auriane Quoix.

19. L. Burnard, *Qu'est-ce que la Text Encoding Initiative ?...*, p. 95.

20. Cet encodage est développé au chapitre 7.5.

2.2.2 Un modèle général pour encoder les catalogues

Les résultats probants des travaux menés par Lucie Rondeau du Noyer et la volonté de Béatrice Joyeux-Prunel de participer aux recherches sur l'encodage automatique des catalogues ont poussé les équipes travaillant sur *GROBID-dictionaries* à repenser l'outil de façon à ce qu'il soit plus adapté aux catalogues. Pour cela, ils souhaitent créer *GROBID-cat*, un *fork* de *GROBID-dictionaries*, dont le modèle serait optimisé et l'encodage XML-TEI plus pertinent pour ce type de sources²¹. Afin de décrire en XML-TEI les entrées des catalogues, Laurent Romary a proposé de créer trois nouvelles balises : <catalogueEntry>, <catalogueDesc> et <catalogueItem>. Celles-ci permettraient d'obtenir un encodage plus approprié qui révélerait mieux la structure des documents.

Réflexions sur la modélisation

À partir de ces premières réflexions, il m'a été demandé d'approfondir la question de la modélisation en XML-TEI de *GROBID-cat* à travers la proposition d'un encodage général – adapté aux niveaux d'encodage de *GROBID-dictionaries*²² – pour les catalogues de vente et d'exposition et les annuaires. Ces trois types de sources ont la particularité d'être structurés de la même manière : ils sont composés d'une liste d'entrées, elles-même divisées en deux parties distinctes : une partie plutôt nominative (présentant l'objet de l'entrée) et une partie descriptive (listant et/ou décrivant les composantes de l'entrée). De là, il est possible de proposer un modèle commun correspondant aux trois premiers niveaux de *GROBID-dictionaries* avec les balises suivantes :

- *page segmentation* : <headnote>, <body>, <footnote>.
- *body segmentation* : <catalogueEntry>.
- *entry segmentation* : <num>, <catalogueDesc>, <catalogueItem>, <note>.

Néanmoins, le contenu des deux niveaux suivants (décrivant le <catalogueDesc> et le <catalogueItem>) varie légèrement d'un type de catalogue à un autre. En effet, les informations retrouvées dans un catalogue d'exposition ne sont pas exactement les mêmes que celles d'un catalogue de vente de manuscrits ou d'un annuaire. Le <catalogueDesc> des catalogues d'exposition et de vente de manuscrits peut contenir <name> et <trait>, tandis qu'il serait plus approprié d'utiliser les balises <name> et <desc> pour les annuaires. Concernant le <catalogueItem>, seule la balise <desc> est commune aux trois catalogues ; les items des catalogues d'exposition étant plutôt composés de <num>, <title>

21. Mohamed Khemakhem, Simon Gabay, Béatrice Joyeux-Prunel, Laurent Romary, Léa Saint-Raymond et Lucie Rondeau du Noyer, « Information Extraction Workflow for Digitised Entry-based Documents », dans *DARIAH Annual event 2020*, Zagreb, 2020, <https://hal.archives-ouvertes.fr/hal-02508549> (visité le 30/07/2020).

22. Le fonctionnement général de *GROBID-dictionaries* est présenté au chapitre suivant.

et `<desc>`, ceux des catalogues de vente de manuscrits de `<desc>` et `<note>` et ceux des annuaires de `<num>`, `<name>` et `<desc>`.

Les annuaires étant un type de catalogue à part, il a été décidé de ne pas les prendre en compte dans le modèle général pour *GROBID-cat* afin d'encoder les deux derniers niveaux de la manière suivante :

- *desc segmentation* : `<name>`, `<trait>`.
- *item segmentation* : `<num>`, `<title>`, `<desc>`, `<note>`.

Les entrées concernant généralement des personnes (auteur, artiste, collectionneur...), nous avons opté pour la balise `<trait>` dans `<catalogueDesc>` car elle permet d'insérer des informations biographiques. Les balises `<title>`, `<desc>` et `<note>` du `<catalogueItem>` servent à nommer et décrire les objets.

Exemples

Le premier niveau de *GROBID-dictionaries* segmente la page (*page segmentation*) : il sépare le corps de texte (`<body>`) de l'en-tête (`<headnote>`) et du pied de page (`<footnote>`). Ensuite, le deuxième niveau (*body segmentation*) segmente le corps de texte en entrées (`<catalogueEntry>`), ici en bleu. Au niveau de l'*entry segmentation* sont trouvés le numéro de l'entrée (`<num>`) en jaune, le nom de l'auteur accompagné ou non d'informations biographiques (`<catalogueDesc>`) en rouge, et le ou les objets présent·e·s dans l'entrée (`<catalogueItem>`) en vert. Le quatrième niveau de segmentation (*desc segmentation*) permet d'affiner l'encodage au niveau du `<catalogueDesc>`. Y est distingué le nom (`<name>`) en violet, des informations biographiques (`<trait>`) en blanc. Enfin, le cinquième niveau (*item segmentation*) permet d'affiner l'encodage au niveau du `<catalogueItem>`. Il peut contenir un numéro (`<num>`) en marron, un titre (`<title>`) en bleu clair, une description (`<desc>`) en orange et des notes (`<note>`) en rose.

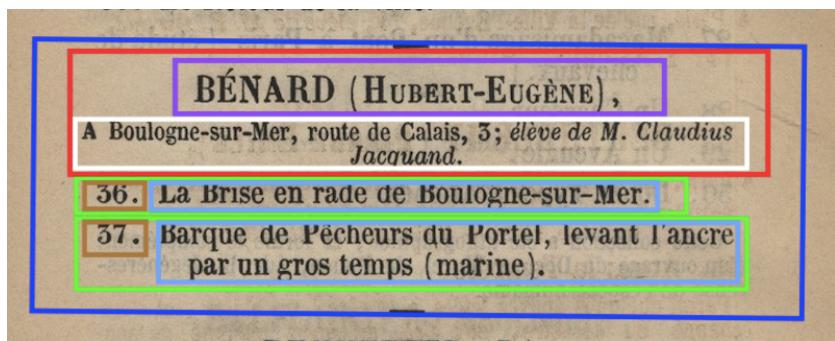


FIGURE 2.2 – Exemple des différents niveaux d'une entrée de catalogue d'exposition.

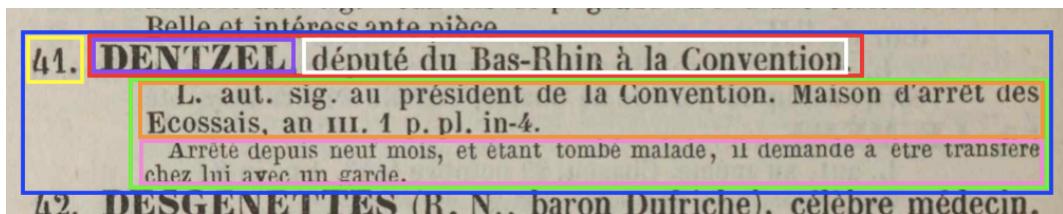


FIGURE 2.3 – Exemple des différents niveaux d'une entrée de catalogue de vente de manuscrits.

Les exemples précédents seraient encodés de la manière suivante en XML-TEI par *GROBID-cat* :

```
<body>
  <catalogueEntry>
    <catalogueDesc>
      <name>BÉNARD (HUBERT-EUGÈNE)</name>,
      <trait>A Boulogne-sur-Mer, route de Calais, 3; élève de
        M. Claudio Jacquand</trait>.
    </catalogueDesc>
    <catalogueItem>
      <num>36</num>.
      <title>La Brise en rade de Boulogne-sur-Mer</title>
    </catalogueItem>
    <catalogueItem>
      <num>37</num>.
      <title>Barque de Pêcheur du Portel, levant l'ancre par
        un gros temps (marine).</title>
    </catalogueItem>
  </catalogueEntry>
</body>
```

FIGURE 2.4 – Exemple d'une entrée de catalogue d'exposition encodée.

```
<body>
  <catalogueEntry>
    <num>41</num>.
    <catalogueDesc>
      <name>DENTZEL</name>,
      <trait>député du Bas-Rhin à la Convention</trait>.
    </catalogueDesc>
    <catalogueItem>
      <desc>L. aut. sig. au président de la Convention. Maison
```

```
d'arrêt des Ecossais, an III. 1 p. pl. in-4.</desc>
<note>Arrêté depuis neuf mois, et étant tombé malade, il
demande à être transféré chez lui avec un garde.</note>
</catalogueItem>
</catalogueEntry>
</body>
```

FIGURE 2.5 – Exemple d'une entrée de catalogue de vente de manuscrits encodée.

La modélisation de l'encodage XML-TEI des catalogues d'exposition est une étape préliminaire essentielle avant l'utilisation de logiciels de *machine learning*. Elle permet de proposer un encodage qui répond aux besoins du projet – dans le cas d'Artl@s, les entrées des catalogues doivent être encodées de manière à ce que les données puissent être récupérées afin de les verser dans BasArt – et d'envisager les balises qui peuvent correspondre aux niveaux de *GROBID-dictionaries*. D'autre part, le travail de modélisation d'un encodage général aux différents types de catalogues a favorisé la conceptualisation du code envisagé pour les différents niveaux de *GROBID-dictionaries*. Il fédère également les précédentes recherches réalisées avec l'outil et permettra d'aboutir à la normalisation de l'encodage des sources catalogographiques.

Deuxième partie

De l'OCR à *GROBID-dictionaries* :
le défi des fichiers ALTO

Chapitre 3

Enjeux de l'intégration des fichiers ALTO dans *GROBID-dictionaries*

Afin d'automatiser la récupération des données issues des catalogues d'exposition, l'équipe d'Artl@s s'est rapidement tournée vers le travail mené par Mohamed Khemakhem. Les résultats de *GROBID-dictionaries* sur l'encodage de documents PDF (catalogues et dictionnaires) étant bons, l'outil de *machine learning* est apparu comme une des solutions possibles pour Artl@s. Ce stage a permis de le tester et d'évaluer les résultats obtenus à partir des catalogues d'exposition afin de les comparer avec d'autres sources.

Pour extraire les données des dictionnaires et des catalogues, *GROBID-dictionaries* s'appuie sur des *features*. Jusqu'alors elles s'attachent à la mise en page du document ainsi qu'à la présence de la ponctuation ; ce sont par exemple des caractères spéciaux et des retours à la ligne. Toutefois d'autres *features*, se basant sur l'information typographique, peuvent être prises en compte par *GROBID-dictionaries* tels que le gras, l'italique ou encore la taille de la police. Mohamed Khemakhem travaille depuis plusieurs mois à l'intégration de ces nouvelles informations qui peuvent notamment être contenues dans des fichiers ALTO (*Analysed Layout and Text Object*).

3.1 *GROBID-dictionaries* : fonctionnement et enjeux

3.1.1 Fonctionnement général

GROBID-dictionaries est un sous-projet de GROBID codé en java qui permet d'encoder automatiquement des documents structurés tels que les dictionnaires, les bibliographies et les catalogues. Il parse, extrait et structure les informations textuelles d'une entrée grâce à un modèle de *machine learning* CRF¹, ce qui fait que l'outil est indépen-

1. En *machine learning*, un modèle CRF est un modèle d'apprentissage supervisé, c'est-à-dire que nous donnons à la machine des exemples annotés sur lesquels s'appuyer. L'intelligence artificielle apprend à partir des exemples et ajuste ses paramètres en fonction.

dant de tout modèle de structuration². Par ailleurs, le modèle CRF permet d'entraîner des modèles à partir de petits jeux de données, ce qui est impossible avec d'autres outils de *machine learning*. Néanmoins, il généralise difficilement, ce qui explique la nécessité de créer des modèles adaptés à chaque typologie de catalogue. Pour encoder les documents en XML-TEI, *GROBID-dictionaries* utilise un modèle en cascade découplant un document en cinq niveaux³ :

- *dictionary segmentation* : ce premier niveau sépare le corps de texte (`<body>`, des titres et numéros de pages situés en entête (`<headnote>`) et en pied de page (`<footnote>`)).
- *dictionary body segmentation* : il segmente les différentes entrées (`<entry>`) contenues dans le `<body>`.
- *lexical entry* : ce troisième niveau distingue les différentes composantes d'une entrée, chacune pouvant être séparée en plusieurs blocs⁴. Pour les besoins des catalogues d'exposition, seules les balises `<lemma>`, `<sense>` et `<dictScrap>` ont été utilisées. Dans `<lemma>` sont regroupées les informations concernant l'artiste et dans `<sense>` celles propres à une œuvre. La balise `<dictScrap>` permet d'ajouter les informations non classables.
- *form* : ce niveau est divisé en deux parties séparant le nom de l'exposant de sa biographie : `<name>` et `<desc>`.
- *sense* : ce dernier niveau segmente l'item d'une entrée en deux ou trois parties selon les informations disponibles : `<num>` (numéro de l'œuvre exposée), `<subSense>` (titre de l'œuvre) et `<note>` (informations complémentaires sur l'œuvre).

Il est possible d'encoder un document encore plus finement, en allant au-delà des cinq premiers niveaux de *GROBID-dictionaries*. Cependant, en fonction du type de source, il peut être difficile à l'outil de segmenter à nouveau des informations. En effet, *GROBID-dictionaries* s'appuie sur des *features*⁵, les informations lui permettant de segmenter chaque niveau ; or certaines informations comme la technique, les dimensions ou la date de l'œuvre peuvent difficilement être catégorisées par la machine. Celle-ci ne fait pas de différence entre les chiffres d'une date et ceux d'une dimension ; de même qu'elle ne distingue pas une chaîne de caractères d'une autre. Pour contrer cela, il faut faire appel à d'autres outils comme les regex (expressions régulières) ou désambiguïser les entités nommées à l'aide de logiciels de reconnaissance d'entités nommées comme *GROBID-ner*⁶

2. L. Rondeau du Noyer, *Encoder automatiquement des catalogues en XML-TEI...*, p. 34.

3. M. Khemakhem, *Grobid-dictionaries...*

4. Se référer au schéma (Figure 3.1).

5. Si elles ont une représentation physique – sous forme de ponctuation –, elles sont encodées dans une balise `<pc>`.

6. Patrice Lopez, *grobid-ner*, GitHub, <https://github.com/kermitt2/grobid-ner> (visité le 27/08/2020).

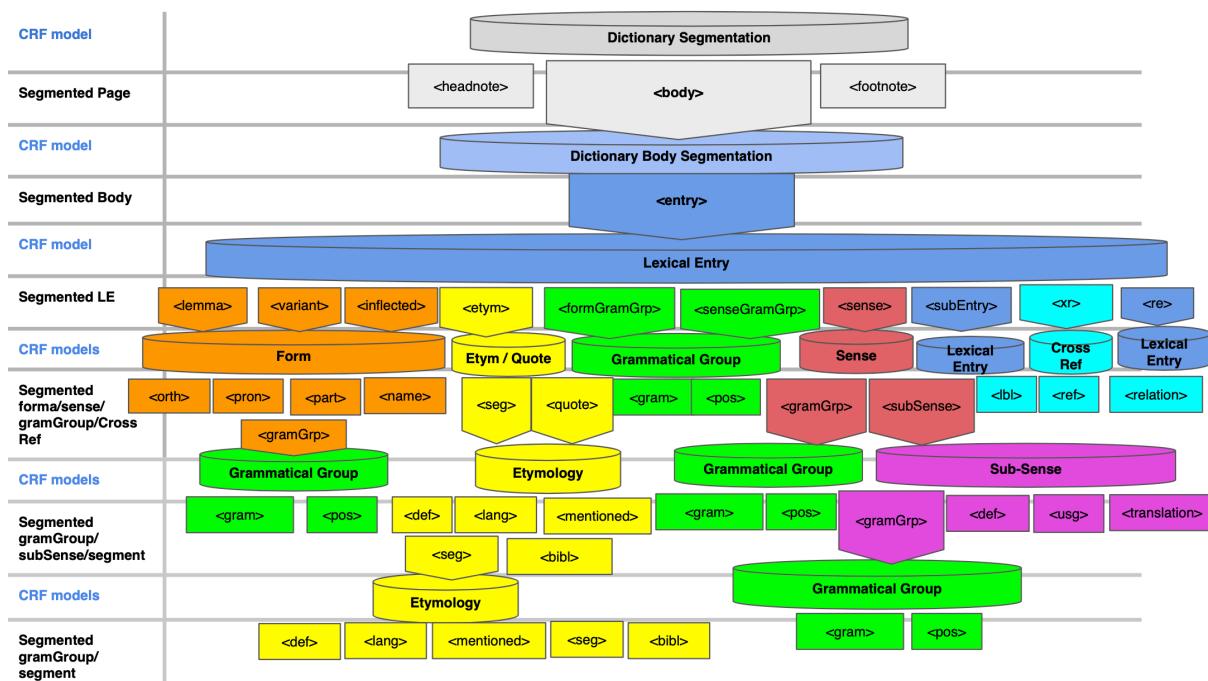


FIGURE 3.1 – Schéma du modèle CRF en cascade de *GROBID-dictionaries*.

Le *workflow* d’annotation de *GROBID-dictionaries* suit la méthode MATTER (*Model - Annotate - Train - Test - Evaluate - Revise*) développée par James Pustejovsky et Amber Stubbs⁷. Elle consiste à développer un modèle CRF adapté à la structure d’un texte et intégré dans l’architecture en cascade du modèle général. Une fois le modèle de *machine learning* construit, chaque bloc de texte est assigné à une balise XML-TEI. Ensuite il est possible d’entraîner un modèle à partir de l’annotation d’un certain nombre de données. Ce modèle est testé afin de vérifier sa fonctionnalité puis les résultats obtenus sont évalués de manière à revoir et améliorer le modèle. Cette méthode cyclique donne ainsi la possibilité d’annoter de nouvelles données pour augmenter l’efficacité du modèle, de corriger les anomalies ou encore d’ajuster les *guidelines* en fonction des retours des utilisateur·rice·s⁸

Pour encoder un document, il est possible d’utiliser le modèle par défaut⁹ accessible via l’interface graphique en ligne de *GROBID-dictionaries*¹⁰ ou de générer des données d’entraînement en utilisant le terminal et des lignes de commande. Dans le second cas, l’utilisateur·rice peut utiliser un modèle déjà entraîné, disponible en ligne¹¹ ou entraîner

7. M. Khemakhem, A. Herold et L. Romary, « Enhancing Usability for Automatically...», p.3.

8. *ibid.*

9. Ce modèle par défaut permet uniquement d’encoder des dictionnaires.

10. L’interface graphique de *GROBID-dictionaries* est disponible à cette adresse : <https://traces1.inria.fr/grobid-dictionaries/>.

11. Des modèles entraînés à partir de dictionnaires sont disponibles sur le GitHub de Mohamed Khemakhem (<https://github.com/MedKhem/grobid-dictionaries>), d’autres entraînés à partir de catalogues de vente de manuscrits sont disponibles sur le GitHub de Katabase (https://github.com/katabase/GROBID_Dictionaries) et ceux entraînés à partir de catalogues d’exposition sont dispo-

un modèle localement. Dans le premier cas, l’utilisateur·rice peut simplement lancer l’entraînement des différents niveaux de *GROBID-dictionaries* d’un modèle avant d’accéder à l’interface graphique en local pour y charger des documents ou entraîner de nouvelles données afin d’améliorer le modèle.

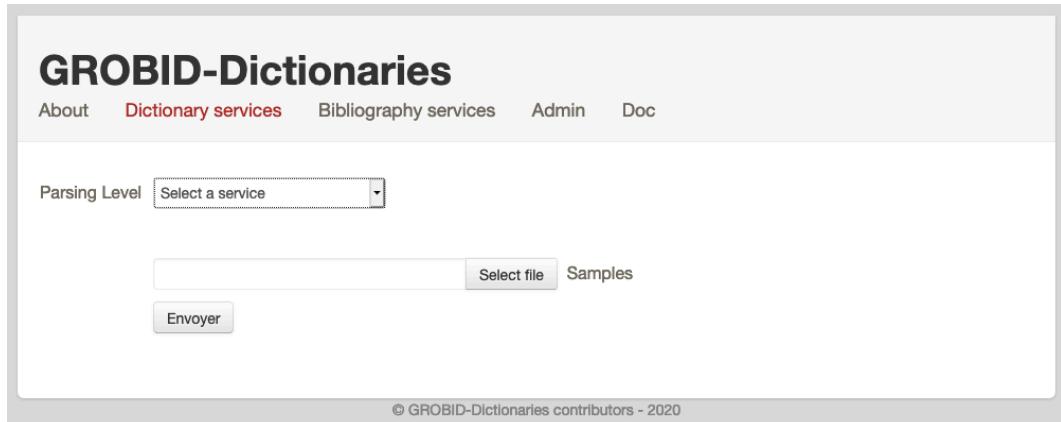


FIGURE 3.2 – Capture d’écran de l’interface graphique de *GROBID-dictionaries*, août 2020.

Pour annoter de nouvelles données¹², il faut – après avoir préalablement placé le ou les document·s à entraîner dans le dossier prévu à cet effet et synchronisé son dossier de travail avec l’application – passer par le terminal et lancer des lignes de commande¹³ qui créent des données d’entraînement à chaque niveau. Pour chacun d’entre eux, *GROBID-dictionaries* génère cinq fichiers :

- un fichier XML-TEI, à annoter manuellement ;
- un fichier .rawtext contenant le texte brut ;
- un fichier renseignant les *features* ;
- un schéma Relax NG décrivant les différentes balises autorisées ;
- une feuille de style CSS permettant d’utiliser un éditeur XML, comme Oxygen, en mode « auteur »¹⁴.

Il faut ensuite annoter le fichier XML-TEI en encodant le texte dans les balises correspondantes via le mode « auteur » de l’éditeur XML. Une fois annoté, tous les fichiers générés par *GROBID-dictionaries* doivent être déplacés dans des dossiers spécifiques¹⁵ de manière à pouvoir lancer l’entraînement. Il faut au préalable vérifier que le processus

nibles sur le GitLab de Béatrice Joyeux-Prunel (<https://gitlab.unige.ch/Beatrice.Joyeux-Prunel/visual-contagions/-/tree/master/Catalogues>).

12. Un exemple précis est décrit au chapitre 6.

13. Les lignes de commandes sont détaillées en annexe B.

14. L. Rondeau du Noyer, *Encoder automatiquement des catalogues en XML-TEI...*, p. 39.

15. Des données doivent être placées dans le dossier dédié à l’entraînement et d’autres dans celui dédié à son évaluation.

d'annotation fonctionne bien en choisissant les mêmes données d'entraînement et d'évaluation : les scores obtenus doivent être égaux à 100%¹⁶. À la fin de l'entraînement, si les résultats sont bons, c'est-à-dire supérieurs à 90%, le serveur local peut être lancé.

3.1.2 Intérêts et enjeux

Le principal intérêt de *GROBID-dictionaries* est de faire gagner du temps aux chercheur·se·s en encodant automatiquement des documents structurés car l'encodage manuel est chronophage. L'outil permet de récupérer un fichier en partie encodé, qu'il faut relire et dont il faut compléter les métadonnées afin de corriger les éventuelles erreurs d'encodage. D'autre part, son développement participe aux recherches sur le *machine learning* et les intelligences artificielles dans le domaine des humanités numériques. Il sert également la recherche sur l'encodage et la structuration des textes avec la TEI.

L'enjeu de *GROBID-dictionaries* est de pouvoir améliorer son efficacité en ajoutant de nouvelles *features* via l'intégration des fichiers ALTO dans son fonctionnement ; puis de l'évaluer, ce qui fût l'objet de ce stage. Il importe également de tester l'outil sur un autre type de catalogues comme les catalogues d'exposition et de l'intégrer dans un *workflow* visant à traiter un catalogue d'un format image ou PDF jusqu'à son intégration dans une base de données.

3.2 Intérêt du format ALTO

3.2.1 Le format ALTO

Brève présentation du standard

Le standard ALTO est un format qui permet de stocker des informations sur la mise en page et l'OCR de tout type de document imprimé. Il suit les recommandations du langage XML afin de conserver l'information sur la mise en page et le texte dans des attributs rattachés à une balise décrivant un bloc de texte, une ligne ou un mot. Y est notamment renseigné la position sur la page d'un caractère, d'un mot, d'un paragraphe, d'une illustration ou encore d'une note de bas de page¹⁷. Ce standard est principalement utilisé par les bibliothèques qui l'associent au standard METS (*Metadata Encoding and Transmission Standard*). De fait, un fichier ALTO est utilisé comme une extension d'un fichier METS, qui contient les métadonnées d'un document tandis que l'ALTO stocke l'information physique de ce document. La Library of Congress et la BnF l'utilisent principalement pour stocker le texte OCRisé de périodiques.

16. M. Khemakhem, A. Herold et L. Romary, « Enhancing Usability for Automatically..., p. 5.

17. Frederick Zarnndt, Joachim Bauer, Markus Enders, Brian Geiger, Kia Siang Hock et Jukka Kervinen, « The ALTO Editorial Board : Collaboration and Cooperation across Borders », dans *IFLA World Library and Information Congress*, Singapore, 2013, p. 2.

À l'origine, il a été développé lors du projet METAe¹⁸ pour les besoins du standard METS, créé pour la Library of Congress. L'un de ses développeurs, Claus Gravenhorst explique lors d'une interview la raison qui les a poussés à créer cette norme :

Au cours du projet METAe, nous avons appris qu'il n'existe aucun standard gérant la position des mots et l'information contenue dans la mise en page (espaces, marges, etc.), une caractéristique pourtant essentielle pour les référentiels capables de mettre en évidence les éléments des documents. C'est pourquoi le schéma ALTO a été développé. Dans les fichiers METS, il y a un pointeur vers les fichiers ALTO qui contiennent le texte, d'autres éléments (illustrations, etc.) et la position des mots. Nous aimerais qu'ALTO ou un schéma similaire devienne un standard puisque nous n'y voyons aucune alternative pour l'instant.¹⁹

Depuis 2009, le schéma est maintenu par la Library of Congress qui édite les nouvelles versions et aide à favoriser son usage dans les bibliothèques.

Version utilisée

Nous utilisons la version 3.0 d'ALTO²⁰ datant d'août 2014, qui spécifie le schéma XML dans la balise suivante :

```
<alto xmlns="http://www.loc.gov/standards/alto/v3/alto.xsd"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.loc.gov/standards/alto/v3/alto.xsd
                           http://www.loc.gov/standards/alto/v3/alto.xsd ">
```

Chaque fichier ALTO est composé de trois sections :

- <Description> décrit les caractéristiques du fichier : l'unité de mesure utilisée (pixels, millimètres, pouces), sa source, les informations concernant l'OCR...
- <Styles> liste les différents styles typographiques (gras, italique...) utilisés dans le document. Ils peuvent être ajoutés manuellement dans des balises <TextStyle>.

18. Un projet de recherche européen, financé sur trois ans, qui a regroupé 14 partenaires venants de 7 pays européens et des États-Unis.

19. « *During the METAe project, we learned that there is no standard to handle word positions and physical layout information (print space, margins, etc.), an essential feature for high performance repositories that are able to highlight elements within documents. Therefore, the ALTO schema has been developed. In the METS file, there are file pointers to the ALTO files that contain the text, other elements (illustrations, etc.), and word positions. We would like ALTO or a similar schema to become a standard as we do not see an alternative right now.* » dans ALTO : Technical Metadata for Layout and Text Objects (Standards, Library of Congress), <https://www.loc.gov/standards/alto/about.html> (visité le 31/08/2020).

20. La documentation de cette version est disponible à l'adresse suivante : <https://www.loc.gov/standards/alto/v3/alto-3-0.xsd>.

— <Layout> renseigne le contenu de la page qui est divisée en plusieurs régions (corps de texte, marges, pied de page...). C'est dans cette partie que sont renseignées les informations sur la mise en page et le contenu textuel.

Chaque <Page> contient une balise <PrintSpace> composée de <TextBlock> qui correspondent à un paragraphe. Ils sont divisés en <TextLine> (ligne), elles-mêmes décomposées en <String> (mot). Chacune de ces balises comprend les attributs suivants : @ID (identifiant), @HEIGHT (hauteur), @WIDTH (largeur), @VPOS (position verticale), @HPOS (position horizontale). De plus, <TextLine> est complétée par l'attribut @BASELINE, qui renseigne le point inférieur gauche du début de la ligne, et <String> est complétée par les attributs @CONTENT, précisant le mot, et @STYLEREFS, qui renvoie au style typographique précédemment défini dans <Styles>.

```
<Page ID="Page1" PHYSICAL_IMG_NR="1" HEIGHT="2200" WIDTH="1101">
  <PrintSpace HEIGHT="2162" WIDTH="1032" VPOS="20" HPOS="23">
    <TextBlock ID="r_1_1" HEIGHT="27" WIDTH="17" VPOS="25" HPOS="531">
      <TextLine ID="tl_1" BASELINE="61" HEIGHT="56" WIDTH="28" VPOS="5" HPOS="526">
        <String HEIGHT="56" WIDTH="84" VPOS="5" HPOS="470" CONTENT="8" ID="tl_1_1" STYLEREFS="FONT0"/>
      </TextLine>
    </TextBlock>
    <TextBlock ID="r_2_1" HEIGHT="31" WIDTH="214" VPOS="118" HPOS="434">
      <TextLine ID="tl_2" BASELINE="172" HEIGHT="80" WIDTH="225" VPOS="92" HPOS="428">
        <String HEIGHT="80" WIDTH="263" VPOS="92" HPOS="391" CONTENT="AUTOGRAPHES." ID="tl_2_1" STYLEREFS="FONT0"/>
      </TextLine>
    </TextBlock>
    <TextBlock ID="r_3_1" HEIGHT="69" WIDTH="994" VPOS="207" HPOS="54">
      <TextLine ID="tl_3" BASELINE="245" HEIGHT="73" WIDTH="1005" VPOS="172" HPOS="49">
        <String HEIGHT="73" WIDTH="91" VPOS="172" HPOS="34" CONTENT="1." ID="tl_3_1" STYLEREFS="FONT0"/>
        <SP HEIGHT="73" WIDTH="15" VPOS="172" HPOS="125"/>
        <String HEIGHT="73" WIDTH="198" VPOS="172" HPOS="79" CONTENT="BERNARDIN" ID="tl_3_2" STYLEREFS="FONT0"/>
        <SP HEIGHT="73" WIDTH="15" VPOS="172" HPOS="277"/>
        <String HEIGHT="73" WIDTH="91" VPOS="172" HPOS="232" CONTENT="DE" ID="tl_3_3" STYLEREFS="FONT0"/>
        <SP HEIGHT="73" WIDTH="15" VPOS="172" HPOS="323"/>
        <String HEIGHT="73" WIDTH="259" VPOS="172" HPOS="277" CONTENT="SAINT-PIERRE," ID="tl_3_4" STYLEREFS="FONT0"/>
        <SP HEIGHT="73" WIDTH="15" VPOS="172" HPOS="536"/>
        <String HEIGHT="73" WIDTH="244" VPOS="172" HPOS="491" CONTENT="littérateur." ID="tl_3_5" STYLEREFS="FONT0"/>
        <SP HEIGHT="73" WIDTH="15" VPOS="172" HPOS="734"/>
        <String HEIGHT="73" WIDTH="259" VPOS="172" HPOS="689" CONTENT="Paris" ID="tl_3_6" STYLEREFS="FONT2"/>
        <SP HEIGHT="73" WIDTH="15" VPOS="172" HPOS="947"/>
        <String HEIGHT="73" WIDTH="91" VPOS="172" HPOS="902" CONTENT="26" ID="tl_3_7" STYLEREFS="FONT0"/>
        <SP HEIGHT="73" WIDTH="15" VPOS="172" HPOS="993"/>
        <String HEIGHT="73" WIDTH="122" VPOS="172" HPOS="932" CONTENT="novem" SUBS_TYPE="HypPart1" SUBS_CONTENT="novembre" ID="tl_3_8" STYLEREFS="FONT0"/>
        <HYP CONTENT="-"/>
      </TextLine>
    </TextBlock>
```

FIGURE 3.3 – Capture d'écran d'un fichier ALTO tiré d'un catalogue de vente de manuscrits, août 2020.

Les mesures contenues dans les attributs de position sont par défaut indiquées en 1/10^e de millimètre ou en 1/1200^e de pouces. Elles peuvent être converties en pixels grâce à la formule suivante pour les millimètres :

$$pixel = value * resolution / 254$$

et à celle-ci pour les pouces :

$$pixel = value * resolution / 1200$$

Pour les besoins de *GROBID-dictionaries*, nous utilisons les pixels, ce qui permet d'utiliser les coordonnées des fichiers ALTO indépendamment de la résolution. La conversion en pixels est généralement opérée par les logiciels d'OCR comme Transkribus. L'équipe de *GROBID-dictionaries* s'est tournée vers le format ALTO-XML plutôt que le format PAGE-XML (*Page Analysis and Ground truth Elements*) car ce dernier – même s'il permet de décrire plus finement le contenu d'une page – contient des informations supplémentaires non utilisées dans le projet.

3.2.2 Intérêt de l'ALTO : l'exemple du *Konstruktivisten*

Outre l'apport de nouvelles informations typographiques et propres à la mise en page d'un document – présentées comme des *features* supplémentaires dans *GROBID-dictionaries* – il importe de comprendre l'intérêt qu'offre ce format à travers l'étude de cas du catalogue d'exposition *Konstruktivisten* (1937).

Il a été réalisé par le graphiste suisse Max Bill (1908-1994) pour la *Kunsthalle* de Bâle. Ce dernier a prôné une sobriété visuelle donnant immédiatement accès à l'information²¹. Pour le catalogue *Konstruktivisten*, il a choisi une police minimale sans serif qu'il a décliné dans deux corps distincts : le premier, en gras, est utilisé pour la prose, les noms d'artistes et les titres d'œuvres tandis que le second, non gras, est employé pour les notes et les biographies. L'œil distingue ainsi facilement les deux types d'informations. D'autre part, une importance est accordée à l'emplacement du texte sur la ligne. Par exemple, le nom de l'artiste est plus à gauche que la numérotation des œuvres, comme le remarque Roxane Jubert²², tandis que le nom de la collection dont est issue l'œuvre est ferré à droite. D'après elle, ces détails – si subtils soient-ils – permettent d'identifier plus rapidement les informations. Ils sont d'autant plus importants que Max Bill a fait le choix de supprimer les lettres majuscules et de réduire la ponctuation au minimum²³.

À son image, d'autres graphistes de l'école suisse comme Josef Müller-Brockmann et Emil Ruder ont contribué à l'élaboration des principes du style typographique international dans les années 1930. Les catalogues d'exposition suivant ce style sont marqués par l'emploi de caractères sans serif, de préférence en minuscules et par des rythmes horizontaux et verticaux²⁴. Les différentes informations d'une entrée d'un de ces catalogues sont alors identifiables les unes des autres uniquement à la taille de la police, à l'emploi du gras et à la position des mots sur une ligne. Ces informations peuvent être précisées dans des fichiers ALTO, contrairement au PDF OCRisé qui contient uniquement le texte dénué de toute information typographique et les retours à la ligne.

21. Roxane Jubert, « Entre voir et lire. La conception visuelle des catalogues d'exposition », dans *Cahiers du Musée national d'art moderne*, n°56/57, Paris, 1996, p. 40.

22. *ibid.*, p. 44.

23. Il emploie la ponctuation uniquement dans le texte en prose.

24. R. Jubert, « Entre voir et lire... », p. 48.

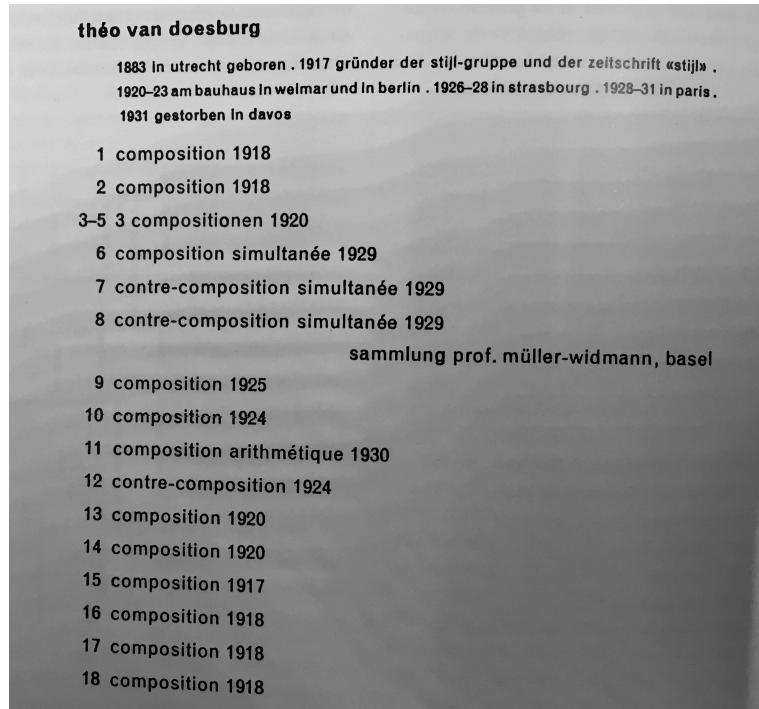


FIGURE 3.4 – Exemple d'entrée du catalogue *Konstruktivisten*, Kunsthalle Basel, 1937.

En définitive, le format ALTO présente un double intérêt : il ajoute des informations supplémentaires contenues dans les *features*, comme le gras, l'italique, la taille de la police, et il donne la possibilité à *GROBID-dictionaries* de pouvoir encoder des catalogues dont la typographie est dans un style minimaliste.

Chapitre 4

Production de fichiers ALTO en sortie de l'OCR

Les fichiers ALTO peuvent être générés par des logiciels d'OCR. Par défaut, un document dont seul le texte est OCRisé, c'est-à-dire sans l'information typographique, permet d'obtenir un fichier ALTO décrivant l'emplacement des lignes, des mots et des images sur la page. Pour obtenir l'information typographique dans l'OCR, il faut réussir à l'insérer dans le texte brut. Il est possible d'y arriver de diverses manières, comme en entraînant un modèle HTR (*Handwritten Text Recognition*)¹ qui reconnaît le gras et l'italique et les met en évidence. Le texte amélioré de l'OCR est ensuite retranscrit dans les fichiers ALTO. Ceux-ci doivent subir de légères modifications afin d'être conformes au format traité par *GROBID-dictionnaries*. Ces deux étapes ont été pensées par Simon Gabay et Ljudmila Petkovic, chargée de recherche pour le projet e-dtiones. Cette dernière a écrit deux scripts python, un premier qui évalue le modèle HTR et un second qui corrige l'OCR et transforme les fichiers ALTO.

4.1 Conservation de l'information typographique dans l'OCR

4.1.1 Entraînement d'un modèle HTR

Pour conserver l'information typographique, Simon Gabay et Ljudmila Petkovic ont eu l'idée d'entraîner un modèle HTR qui reconnaît les mots en gras et en italique puis les encode dans des balises `` et `` pour le gras et des balises `<i>` et `</i>` pour l'italique. Ils ont également pensé à entourer ces mots par d'autres signes comme des caractères spéciaux ou exotiques, ou encore d'encoder chaque mot entre une succession

1. Un modèle HTR est plus efficace qu'un modèle OCR pour distinguer les variances typographiques d'un imprimé. Voir L. Rondeau du Noyer, *Encoder automatiquement des catalogues en XML-TEI...*, p. 31.

de trois lettres : **bbb** pour un mot en gras, **iii** pour un mot en italique, **ccc** pour un mot en gras et italique et **nnn** pour un mot dénué d'information typographique². Néanmoins, ils n'ont pas testé ces dernières manières de souligner le gras et l'italique, ce qui aurait été intéressant afin de comparer l'efficacité de chaque modèle.

Le modèle HTR, nommé TYPO_18_04_2020, a été entraîné sur Transkribus par Simon Gabay. Le choix du logiciel d'OCR s'est rapidement tourné vers Transkribus car, outre son interface relativement simple d'utilisation, il permet de créer des collections partagées et d'entraîner des modèles à partir d'un petit jeu de données annotées. Par exemple, le modèle TYPO_18_04_2020 s'appuie sur environ 300 pages annotées issues de catalogues de vente de manuscrits et de catalogues d'exposition principalement édités au XIX^e siècle.

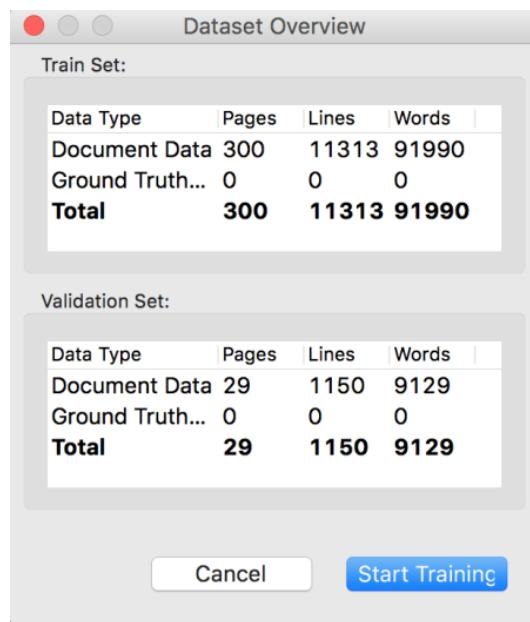


FIGURE 4.1 – Capture d'écran du jeu de données du modèle TYPO_18_04_2020 entraîné dans Transkribus, juin 2020.

Afin d'améliorer le modèle, j'ai été chargée de produire des données d'entraînement à partir de catalogues d'exposition. Chacun d'entre eux fait l'objet d'un double traitement dans Transkribus : la fonction OCR (Abbyy FineReader) reconnaît la mise en page du document et propose une première couche de texte, puis, le modèle TYPO_18_04_2020 ajoute les balises **** et **<i>** à la transcription. Durant cette phase d'annotation, il était important de veiller à la bonne formation des balises, mais aussi des *baselines*³.

La création de ce modèle, dont le taux d'erreur est de 0.89%, a permis de contrer un problème soulevé l'année dernière par Lucie Rondeau du Noyer : jusqu'alors, Transkribus

2. Simon Gabay et Ljudmila Petkovic, *19th fixed-price and auction catalogues : GroundTruth and Models for OCR*, GitHub, <https://github.com/ljpetkovic/OCR-cat> (visité le 01/09/2020).

3. Une *baseline* correspond à la ligne allant du coin inférieur gauche du premier caractère au coin inférieur droit du dernier caractère d'une ligne.

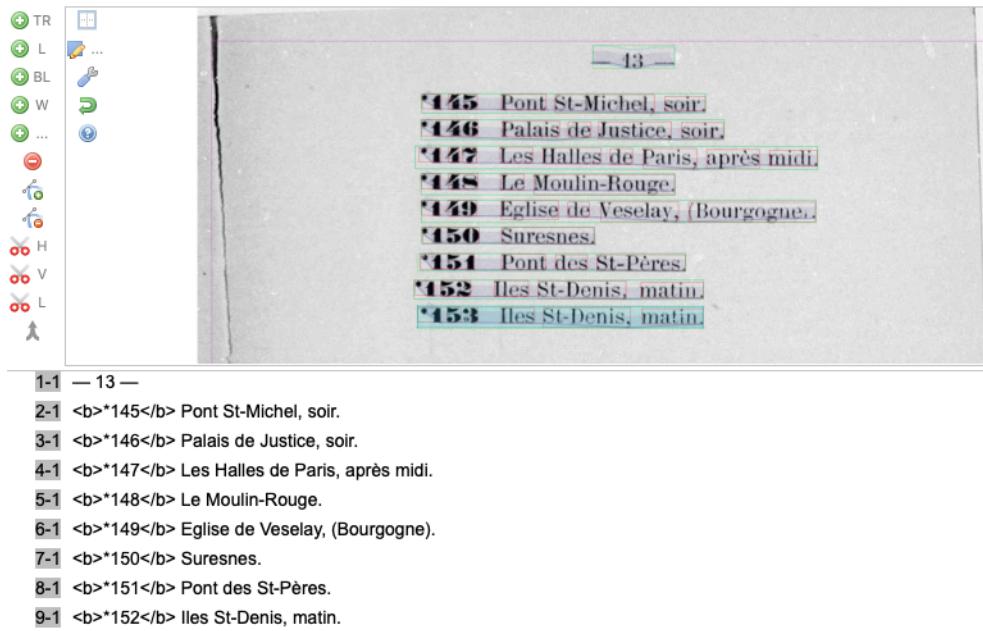


FIGURE 4.2 – Capture d'écran de la transcription avec les balises **** d'un catalogue d'exposition dans Transkribus, août 2020.

ne pouvait pas indiquer automatiquement dans le résultat exporté les passages imprimés en gras et en italique⁴. Cependant, le modèle a uniquement été entraîné sur des catalogues en langue française, or Artl@s travaille sur des sources imprimées en diverses langues, notamment l'anglais, l'italien et le portugais. Pour contrer cela, il faudrait tester le modèle afin de voir s'il arrive à généraliser la reconnaissance des mots, du gras et de l'italique sans accorder de l'importance à la langue ou s'il faut au contraire créer des modèles spécifiques à chaque langue⁵.

4.1.2 Évaluation du modèle

De manière à vérifier l'efficacité du modèle et la bonne formation des balises, Ljudmila Petkovic a écrit un script python itérant sur un fichier .txt contenant la transcription d'un catalogue qui recherche, à l'aide d'expressions régulières, les balises ****, ****, **<i>** et **</i>**. Le script récupère ainsi chaque ligne produite par l'OCR et recherche la présence de balises. Si la ligne en contient, il vérifie ensuite que chaque balise ouvrante est bien suivie d'une balise fermante. Si cela n'est pas le cas, le script cherche à identifier l'erreur de la malformation des balises. En fonction du type d'erreur, il propose une correction qui peut être gérée automatiquement par la machine ou indique en capitales l'origine du problème. Il est possible de faire corriger automatiquement une balise ouvrante mal fermée si elle est bien suivie d'une balise fermante, par exemple **>13 à 19** est rectifié de la manière

4. L. Rondeau du Noyer, *Encoder automatiquement des catalogues en XML-TEI...*, p. 31.

5. Les logiciels d'OCR s'appuient sur des dictionnaires de langue et sont plus performants dans certaines langues, comme l'anglais et le français puisqu'ils ont été entraînés sur un nombre important de sources anglophones et francophones.

suivante 13 à 19. Chaque ligne traitée par le script est associée à un numéro, entre 0 et 4, indiquant la présence ou non de balises et le type d'erreur :

- 0 : ligne sans balise ;
- 1 : ligne où les balises sont bien formées et dans l'ordre, ou qu'elles peuvent être corrigées automatiquement ;
- 2 : ligne où les balises sont bien formées mais inversées (la balise fermante est avant la balise ouvrante) ;
- 3 : ligne où les balises sont bien formées mais il manque une balise ouvrante ou fermante ;
- 4 : ligne où il y a au moins une balise erronée qui ne peut pas être corrigée automatiquement.

| | |
|--|--|
| 5 Canal Saint-Martin, Paris, 1901-1903. | 1 |
| 6 à 12 Marseille, 1904-1905. | 3 BALISES MANQUANTES |
| >13 à 19 Tunisie, 1906-1907. | 1 13 à 19 Tunisie, 1906-1907. |
| 20 à 30 Corse, 1908-1909. | 4 20 à 30 Corse, 1908-1909. |
| b>31 à 36 Belgique-Nieuport, 1910 | 3 31 à 36 Belgique-Nieuport, 1910 |
| 37 à 43 Seine-et-Oise, Champagne, 1911. | 3 37 à 43 Seine-et-Oise, Champagne, 1911. |
| <i>Esquisses</i> | 1 |

FIGURE 4.3 – Capture d'écran de l'évaluation du modèle HTR, septembre 2020.

Enfin, le script propose un résultat statistique comptant le nombre de balises et les erreurs. Il affiche également un pourcentage qui permet d'avoir un aperçu du taux de la bonne formation des balises et du type d'erreur régulièrement rencontré. Les nombreux échanges avec Ljudmila Petkovic sur ce script et les résultats obtenus nous ont donné l'occasion de l'améliorer en produisant de nouvelles statistiques⁶ et en ajoutant de nouvelles expressions régulières pour corriger des erreurs.

Les deux tableaux suivants présentent les résultats du modèle HTR TYPO_18_04_2020 sur des catalogues d'exposition (Table 4.1) et des catalogues de vente de manuscrits (Table 4.2)⁷. La Table 4.1 regroupe le *Catalogue de la Société des Artistes Indépendants* de 1892 (IND_1), celui de 1923 (IND_2), le *Catalogue de l'exposition annuelle du musée de Rouen* de 1853 (ROU_1) et celui de 1888 (ROU_2). Tandis que la Table 4.2 rassemble *L'Amateur d'autographes* de novembre 1845 (LAC_1), celui de janvier 1846 (LAC_2), le *Catalogue de lettres autographes et manuscrits* d'avril 1856 (LAV_1) et celui de novembre-décembre 1860

6. Par exemple, au départ il n'y avait pas de statistique récapitulant le nombre de malformations, toutes erreurs confondues.

7. Ce second tableau a été réalisé par Ljudmila Petkovic à partir des données du projet editiones.

(LAV_2).

Les résultats du modèle HTR montrent qu'il est efficace, notamment sur les catalogues d'exposition pour lesquels le taux de balises mal formées est inférieur à 1,70%. Il est légèrement supérieur – entre 2,50% et 2,85% – pour les *Catalogues de lettres autographes et manuscrits* édités par Auguste Laverdet. Toutefois, le taux de balises mal formées est supérieur à 8% pour les catalogues de *L'Amateur d'autographes* édités par Jacques Charavay. Par ailleurs, le taux de balises non corrigibles automatiquement est inférieur à 0,40% pour les catalogues d'exposition et avoisine les 1,50% pour les catalogues de vente de manuscrits. La majorité des balises sont donc bien formées ou touchées par des erreurs facilement corrigibles par la machine. Les résultats démontrent également que les balises sont toujours formées dans le bon ordre, c'est-à-dire suivant le schéma balise ouvrante, balise fermante.

Les taux révèlent que le modèle est plus efficace pour les catalogues d'exposition, dont la mise en page est plus aérée que celle des catalogues de vente de manuscrits. De plus, les tests ont été réalisés sur un nombre de ligne plus important pour les catalogues d'exposition (11738 lignes de plus). Un entraînement plus conséquent, comprenant plus de données issues de catalogues de vente de manuscrits, pourrait améliorer les résultats du modèle HTR.

| Type de balise | Nombre | | | | % | | | |
|--|--------|-------|-------|-------|-------|-------|-------|-------|
| | IND_1 | IND_2 | ROU_1 | ROU_2 | IND_1 | IND_2 | ROU_1 | ROU_2 |
| Sans balise | 863 | 3502 | 922 | 2784 | 40.48 | 41.21 | 93.99 | 68.96 |
| Bien formée | 1233 | 4882 | 53 | 1205 | 57.83 | 57.46 | 5.40 | 29.85 |
| Mal formée | 36 | 113 | 6 | 48 | 1.69 | 1.33 | 0.61 | 1.19 |
| Non corrigible automatiquement | 9 | 41 | 0 | 19 | 0.42 | 0.48 | 0.00 | 0.47 |
| Corrigible automatiquement | 27 | 72 | 6 | 29 | 1.27 | 0.85 | 0.61 | 0.72 |
| Non corrigible automatiquement | 9 | 41 | 0 | 19 | 0.42 | 0.48 | 0.00 | 0.47 |
| – Mauvais ordre | 0 | 0 | 0 | 0 | 0.00 | 0.00 | 0.00 | 0.00 |
| – Balise manquante | 9 | 41 | 0 | 19 | 0.42 | 0.48 | 0.00 | 0.47 |
| Corrigible automatiquement | 27 | 72 | 6 | 29 | 1.27 | 0.85 | 0.61 | 0.72 |
| – Balise bien corrigée | 19 | 20 | 0 | 20 | 0.89 | 0.24 | 0.00 | 0.50 |
| – Balise bien corrigée, mauvais ordre | 0 | 0 | 0 | 0 | 0.00 | 0.00 | 0.00 | 0.00 |
| – Balise bien corrigée, balise manquante | 3 | 17 | 1 | 1 | 0.14 | 0.20 | 0.10 | 0.02 |
| – Balise bien corrigée, balise vide | 5 | 35 | 5 | 8 | 0.23 | 0.41 | 0.51 | 0.20 |

TABLE 4.1 – Résultats de l'évaluation du modèle HTR (Catalogues d'exposition).

| Type de balise | Nombre | | | | % | | | |
|--|--------|-------|-------|-------|-------|-------|-------|-------|
| | LAC_1 | LAC_2 | LAV_1 | LAV_2 | LAC_1 | LAC_2 | LAV_1 | LAV_2 |
| Sans balise | 227 | 1547 | 317 | 604 | 49.13 | 69.06 | 74.41 | 77.34 |
| Bien formée | 198 | 469 | 98 | 155 | 42.86 | 20.94 | 23.00 | 19.85 |
| Mal formée | 37 | 224 | 11 | 22 | 8.01 | 10.00 | 2.58 | 2.82 |
| Non corrigible automatiquement | 5 | 55 | 5 | 10 | 1.08 | 2.46 | 1.17 | 1.28 |
| Corrigible automatiquement | 32 | 169 | 6 | 12 | 6.93 | 7.54 | 1.41 | 1.54 |
| Non corrigible automatiquement | 5 | 55 | 5 | 10 | 1.08 | 2.46 | 1.17 | 1.28 |
| – Mauvais ordre | 0 | 0 | 0 | 0 | 0.00 | 0.00 | 0.00 | 0.00 |
| – Balise manquante | 5 | 55 | 5 | 10 | 1.08 | 2.46 | 1.17 | 1.28 |
| Corrigible automatiquement | 32 | 169 | 6 | 12 | 6.93 | 7.54 | 1.41 | 1.54 |
| – Balise bien corrigée | 4 | 48 | 3 | 4 | 0.87 | 2.14 | 0.70 | 0.51 |
| – Balise bien corrigée, mauvais ordre | 0 | 0 | 0 | 0 | 0.00 | 0.00 | 0.00 | 0.00 |
| – Balise bien corrigée, balise manquante | 9 | 39 | 2 | 4 | 1.95 | 1.74 | 0.47 | 0.51 |
| – Balise bien corrigée, balise vide | 19 | 82 | 1 | 4 | 4.11 | 3.66 | 0.23 | 0.51 |

TABLE 4.2 – Résultats de l'évaluation du modèle HTR (Catalogues de vente de manuscrits).

4.2 Correction de l'OCR et transformation des fichiers ALTO

4.2.1 Corriger l'OCR

Le script d'évaluation du modèle HTR classe les erreurs de malformation des balises dans deux catégories : celles corrigibles automatiquement et celles qui ne le sont pas⁸. Ces dernières sont à traiter dans un premier temps. Pour l'instant, ces erreurs sont corrigées manuellement directement dans Transkribus. Seules les erreurs de type 3 et 4 doivent être prises en compte. Les erreurs de type 2 n'étant jamais rencontrées, elles ont été écartées, toutefois, si un·e utilisateur·rice croise cette erreur, il ou elle devra également la corriger à la main. Une fois cette étape de relecture de l'OCR réalisée, la transcription est exportée en format ALTO. Dans Transkribus, il est primordial de sélectionner la sortie ALTO par mots, puisque l'autre version créée des ALTO n'allant pas au-delà de la segmentation par ligne.

Il est ensuite possible de corriger les erreurs restantes automatiquement dans le fichier ALTO grâce à un second script python, également écrit par Ljudmila Petkovic, qui itère sur chaque @CONTENT des balises `<String>`. Il y cherche les balises ``, ``, `<i>` et `</i>` et vérifie qu'elles soient bien associées à la balise correspondante. Si l'une d'entre elles s'avère être mal formée, elle est corrigée à l'aide d'expressions régulières. Celles-ci gèrent deux types d'erreur : les balises dites pleines, c'est-à-dire dont la lettre est bien indiquée, et les balises vides, sans lettres. Ces dernières sont corrigées en fonction de la balise précédente ou suivante. Par exemple, `<i>le Capricorne (Décembre</>` est rectifié par `<i>le Capricorne (Décembre</i>`.

Pour l'instant seules ces erreurs peuvent être corrigées automatiquement. Il est difficile de gérer celles où une balise manque puisque la machine, qui n'a accès qu'à du texte brut, ne peut savoir où placer la balise manquante. De même, elle ne sait pas corriger une succession de deux balises vides. Les deux scripts de Ljudmila Petkovic restent cependant d'une grande aide puisque le premier pointe la majorité des erreurs⁹ et indique, grâce à son code chiffré, lesquelles corriger à la main. Le second gère les autres erreurs à l'intérieur des fichiers ALTO et s'occupe également de les transformer afin qu'ils soient conformes à la version lue par *GROBID-dictionaries*.

8. Se référer aux deux tableaux précédents.

9. Certaines erreurs demeurent invisibles pour la machine, notamment lorsque le modèle HTR ne reconnaît pas l'information typographique. Il arrive que certains mots en gras ou en italique ne soient pas encodés dans des balises. Ces erreurs doivent être corrigées manuellement après une relecture attentive de l'OCR.

4.2.2 Insérer l'information typographique dans les fichiers ALTO

Le second script python gère à la fois la correction des balises et la transformation des fichiers ALTO. Celle-ci est opérée en diverses étapes. Tout d'abord, le fichier est associé au schéma de la version 3.0 d'ALTO¹⁰ afin d'en valider la conformité. Ensuite, la partie **<Description>** est complétée : sont renseignés le nom de la source et des informations concernant le logiciel de transformation du fichier. Sont également ajoutés différents styles typographiques dans la balise **<Styles>** :

- FONT0 : aucun style typographique ;
- FONT1 : gras ;
- FONT2 : italique.

Ces trois styles servent à identifier l'information typographique contenue dans chaque mot. En effet, la dernière étape du script consiste à insérer dans chaque balise **<String>** un attribut **@STYLEREFS** qui précise le style typographique du mot. S'il n'est pas entouré de balises, il est associé au style FONT0. Au contraire s'il est encodé dans une balise, il est relié au style FONT1 pour **** et FONT2 pour **<i>**. De cette manière, l'information typographique est conservée, propre à chaque mot et conforme au format ALTO. Par ailleurs, ce rajout d'information dans l'attribut **@STYLEREFS** permet de supprimer les balises **** et **<i>** du texte, qui retrouve ainsi sa forme originale.

Cette étape d'évaluation et de correction de la transcription puis de transformation des fichiers ALTO est charnière dans la chaîne de traitement des documents. Elle permet de s'assurer que l'information typographique est bien intégrée dans les fichiers avant qu'ils puissent être traités par *GROBID-dictionaries*.

10. Les fichiers ALTO en sortie de Transkribus sont associés à la version 2.

Chapitre 5

Retard du projet et recherche de solutions

Le 6 juillet 2020¹, alors que la phase de préparation et de transformation des fichiers ALTO est terminée, Simon Gabay apprend que la nouvelle version de *GROBID-dictionaries* ne sera pas livrée à temps. Face à cet imprévu, il a fallu trouver rapidement des solutions pour pouvoir entraîner des modèles sur *GROBID-dictionaries* afin de pouvoir livrer un *workflow* utilisable et fonctionnel à l'équipe d'Artl@s. L'objectif était de s'appuyer sur le travail réalisé pendant les deux premiers mois du stage pour produire de nouveaux documents PDF, seul type de fichiers accepté par *GROBID-dictionaries*. L'intérêt du précédent travail est d'avoir inclus dans les fichiers ALTO l'information typographique. L'enjeu est donc de la conserver dans les PDF pour que de nouvelles *features* puissent améliorer l'apprentissage de *GROBID-dictionaries*.

5.1 Retard et difficultés du développeur de *GROBID-dictionaries*

Le travail réalisé au sein de l'équipe d'Artl@s dépend en partie de celui des équipes de *GROBID-dictionaries* et d'e-ditiones. Si la majorité des outils et des scripts ont été créés en amont², la dernière version de *GROBID-dictionaries* n'a pas pu être livrée à temps. Ce retard est dû à diverses raisons, la principale étant que Mohamed Khemakhem, son développeur, a éprouvé des difficultés à faire lire les fichiers ALTO par l'outil de *machine-learning*. Lors d'un échange par mails, il m'a donné les précisions suivantes sur la fonctionnalité de lecture de *GROBID-dictionaries* :

GROBID-dictionaries est un sous-module de GROBID dont il hérite des fonc-

1. Cette date correspond environ à la moitié de mon stage.

2. Par exemple, Ljudmila Petkovic avait créé les scripts avant mon arrivée en stage début mai 2020. Ils ont ensuite été ajustés en fonction des tests et des retours que nous avons pu faire dessus.

tionnalités de base comme la lecture des fichiers en entrée. Cette lecture est une phase cruciale pour plusieurs procédures de traitement du texte d'un document en entrée, notamment celle de la transformation du texte en fichiers d'entraînement et sa structuration TEI déclenchée par les services de la facette web de l'outil.

La prise en charge d'un nouveau type de fichier, ALTO au lieu de PDF, a entraîné une révision en profondeur des briques du système pour assurer l'acheminement correct du texte inclus dans les deux types d'entrée. L'obstacle majeur était de contourner les restrictions des fonctionnalités de base dictées par l'implémentation dans le code de GROBID. Le fait que les deux outils soient développés en parallèle, selon des philosophies différentes, a rendu l'intégration de la fonction de lecture des fichiers ALTO, partiellement intégrée dans GROBID, plus sensible.³

Il explique ici que la lecture des fichiers ALTO et leur transformation en fichier XML-TEI s'est avérée être plus délicate puisqu'il a fallu revoir une partie du code de *GROBID-dictionaries*. En effet, la grande différence entre le texte d'un fichier PDF et celui d'un fichier ALTO est qu'il est dans le premier cas sous une forme continue et facilement récupérable; contrairement au texte stocké dans un fichier ALTO, qui est contenu dans un attribut de balise, et associé à d'autres attributs délivrant diverses informations sur chaque mot. Le processus de récupération du texte d'un fichier ALTO peut s'avérer être plus long puisqu'il faut itérer sur chaque balise. Le code a donc dû être revu en profondeur pour qu'il puisse à la fois lire les fichiers ALTO et en récupérer le texte pour le transférer dans un fichier XML-TEI.

L'expérience du retard de livraison d'un outil a mis en exergue la dépendance d'autrui que suscite un projet inscrit dans un cadre inter-institutionnel, tel celui dans lequel s'est déroulé ce stage. Elle démontre également l'importance du respect des délais – dans la mesure du possible – notamment dans le cadre d'un travail de groupe, où chacun·e s'occupe d'une tâche particulière réutilisée par les autres membres de l'équipes. Ce genre d'événement peut mettre en pause tout un projet. Dans le cas d'Artl@s, nous avons cherché des solutions pour contourner le problème en attendant d'avoir la dernière version de *GROBID-dictionaries*. Son autre version, utilisée par Lucie Rondeau du Noyer l'an passé, permet de traiter des PDF et d'entraîner des modèles à partir de ce format. Nous avons donc cherché des solutions qui permettent de conserver dans les PDF l'information typographique contenue dans les ALTO.

3. Citation tirée d'un mail échangé entre Mohamed Khemakhem et moi-même le 14 août 2020.

5.2 Retour vers le PDF : solution pour conserver l'information typographique

5.2.1 Conserver l'information typographiques dans un PDF : solutions envisagées

Face au retard du développeur de *GROBID-dictionaries*, nous avons dû rapidement trouver une solution pour faire avancer le projet et commencer à entraîner des modèles dans *GROBID-dictionaries*. Pour cela, il fallait transformer les fichiers ALTO en PDF, de façon à ne pas perdre le travail réalisé jusqu'alors, tout en y intégrant l'information typographique. Les discussions entre Simon Gabay, Thibault Clérice (responsable du master TNAH à l'ENC et directeur de recherche du présent mémoire), Béatrice Joyeux-Prunel et moi-même sur les possibilités envisageables ont abouties aux solutions suivantes :

- Récupérer le texte transcrit avec les balises **** et **<i>** et transformer le fichier ALTO en PDF grâce à un script python.
- Récupérer le texte transcrit, remplacer les balises **** et **<i>** par des caractères spéciaux, dits exotiques, et transformer le fichier ALTO en PDF grâce à un script python.
- Récupérer le texte transcrit, supprimer les balises **** et **<i>**, souligner leur contenu par des signes souscrits et transformer le fichier ALTO en PDF grâce à un script python.
- Transformer les fichiers ALTO en PDF grâce à XSL-FO.

L'idée était de tester toutes ces solutions pour ensuite évaluer l'efficacité de chacune. Néanmoins, le manque de temps et la nécessité de faire des recherches et des tests pour s'assurer de leur bon fonctionnement m'ont amenée à faire un choix et à écarter certaines solutions envisagées. La principale contrainte des trois premières possibilités était de réussir à transformer les fichiers ALTO en PDF à l'aide d'un script python. Celui-ci devait itérer sur le @CONTENT des balises **<String>** pour récupérer le texte et transformer ou non les balises **** et **<i>** puis convertir l'ALTO en fichier PDF. Pour cela, il fallait à la fois maîtriser la librairie python `lxml`⁴, pour itérer sur un fichier `.xml`⁵, et la librairie `reportlab`⁶, qui permet de générer des documents PDF à partir d'un script python. Par ailleurs, la troisième solution nécessitait de connaître la librairie `unidecode` et la décomposition unicode NFKD⁷. Tandis que la quatrième solution nécessitait de maîtriser les

4. « lxml - XML and HTML with Python », `lxml`, <https://lxml.de/> (visitée le 05/09/2020).

5. Les fichiers ALTO sont des fichiers `.xml`, puisqu'ils dépendent du langage XML.

6. *ReportLab PDF LibraryUser Guide*, London, 2020, <https://www.reportlab.com/docs/reportlab-userguide.pdf> (visité le 05/09/2020).

7. Les caractères unicodes sont normalisés afin d'être utilisables et lisibles par le plus grand nombre. Chaque caractère fait l'objet d'une décomposition canonique qui permet d'effectuer une comparaison

feuilles de style XSL-FO⁸, qui permettent de mettre en page un document .xml avant sa conversion en PDF, et un processeur FO, tel RenderX⁹, qui transforme le fichier XSL-FO en PDF.

5.2.2 Le choix d’XSL-FO

Le choix

Après avoir fait des tests en python et avec XSL-FO, nous avons choisi de nous tourner vers XSL-FO. Cette décision a été prise en fonction de deux facteurs : le temps et l’accessibilité. Tout d’abord, les tests des scripts python se sont avérés être plus longs à réaliser et certains sont restés sans succès après une semaine de travail. Il aurait fallu pousser les recherches, notamment sur la librairie unicode, pour pouvoir faire fonctionner tous les scripts, mais le manque de temps ne m’a pas permis de le faire. La création d’une feuille de style XSL-FO était plus simple dans le sens où il a uniquement fallu comprendre la manière dont FO met en page un document ; les autres instructions sont spécifiées dans des *templates* de la même façon que dans toute autre feuille de style XSL. D’autre part, une des conditions du *workflow* est son accessibilité puisqu’il sera par la suite réutilisé par les membres de l’équipe d’Art1@s. Certain·e·s d’entre elles ou eux sont peu ou non formé·e·s au numérique, il est donc essentiel de prendre en compte cet aspect. Cette étape de transformation étant supplémentaire, il importait de la rendre simple d’utilisation. Cette dernière contrainte a également joué en la faveur d’XSL-FO, dont la transformation est directement lancée via un éditeur XML tel Oxygen, contrairement au script python qui est lancé par le terminal. De plus, il est possible d’appliquer la feuille de style à tous les fichiers contenus dans un dossier grâce à la création d’un projet dans Oxygen.

XSL-FO

XSL-FO est un vocabulaire XSL qui permet de décrire la mise en page d’un document. Il est possible de créer un document XSL-FO grâce à une feuille de style XSL dans laquelle sont spécifiées des règles ou *templates*. Par exemple, pour conserver l’information typographique contenue dans les fichiers ALTO, nous avons créé une règle qui associe à chaque @STYLEREFS le style typographique approprié. Ainsi, le style FONT1 du fichier ALTO est remplacé par une spécification FO qui met en gras un mot, grâce à l’attribut `font-weight="bold"`.

binaire entre le texte d’origine (non normalisé) et son équivalence canonique (normalisée). La décomposition NFKD décompose les caractères par équivalence canonique et de compatibilité et les réordonne. Voir « Normalisation Unicode », Wikipédia, https://fr.wikipedia.org/wiki/Normalisation_Unicode (visité le 05/09/2020).

8. Extensible Stylesheet Language (XSL) Version 1.1, <https://www.w3.org/TR/xsl/#d0e4611> (visité le 30/07/2020)

9. RenderX, *RenderX - Support - XSL Formatting Objects Tutorial - RenderX*, <http://www.renderx.com/tutorial.html> (visité le 30/07/2020).

```

<xsl:template match="*[local-name() = 'String']">
  <xsl:if test="@STYLEREFS = 'FONT0'">
    <fo:inline>
      <xsl:value-of select="@CONTENT"/>
    </fo:inline>
  </xsl:if>
  <!-- add bold or italic -->
  <xsl:if test="@STYLEREFS = 'FONT1'">
    <fo:inline font-weight="bold">
      <xsl:value-of select="@CONTENT"/>
    </fo:inline>
  </xsl:if>
  <xsl:if test="@STYLEREFS = 'FONT2'">
    <fo:inline font-style="italic">
      <xsl:value-of select="@CONTENT"/>
    </fo:inline>
  </xsl:if>
</xsl:template>

```

FIGURE 5.1 – Règle XSL qui applique le style typographique approprié à chaque mot.

Le fichier ALTO est donc transformé une première fois en fichier XSL-FO puis celui-ci est converti en PDF grâce à un processeur FO. Nous avons opté pour le processeur RenderX qui est plus performant que son concurrent Apache FOP. L’information typographique est bien retranscrite dans les PDF, où figurent le gras et l’italique. La seule difficulté a été de rendre compte de la mise en page : la valeur du début de ligne étant propre à chaque page, il était difficile de proposer une variable stable qui puisse prendre en compte cette information. Chaque ligne est ferrée à gauche et perd donc l’information de l’emplacement du début de la ligne et de la taille des caractères.

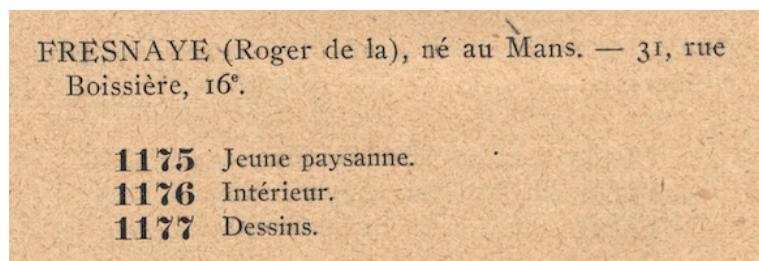


FIGURE 5.2 – Exemple d’entrée du *Catalogue de la 37e exposition 1926... du 20 mars au 2 mai / Société des artistes indépendants*, Société des artistes indépendants, Paris, 1926.

FRESNAYE (Roger de la), né au Mans. — 31, rue

Boissière, 16e.

1175 Jeune paysanne.

1176 Intérieur.

1177 Dessins.

FIGURE 5.3 – Exemple d’entrée du *Catalogue de la 37e exposition 1926* de la Société des artistes indépendants, obtenue avec XSL-FO.

Le retard de Mohamed Khemakhem a engendré une modification du *workflow* qui a été augmenté d’une étape supplémentaire. Ce retour au format PDF a permis d’entraîner des modèles avec *GROBID-dictionaries* en attendant de pouvoir utiliser le logiciel avec les fichiers ALTO. Grâce à la transformation des fichiers ALTO en PDF via XSL-FO, deux types de PDF ont été produits à partir d’un corpus regroupant une sélection de catalogues d’exposition de la Société des artistes indépendants : des catalogues avec information typographique et d’autres sans. De cette manière, nous avons pu préparer des PDF de quelques pages pour la campagne d’entraînement de *GROBID-dictionaries*.

Chapitre 6

Entraînement et évaluation de modèles avec *GROBID-dictionaries*

Un des enjeux de ce stage est l'évaluation des modèles entraînés sur *GROBID-dictionaries*. La comparaison des résultats obtenus à partir des PDF et des ALTO ainsi que des PDF avec et sans l'information typographique va permettre d'évaluer l'efficacité de chaque modèle, ainsi que l'impact de l'ajout de l'information typographique dans les documents. Ces deux paramètres sont essentiels à prendre en compte dans l'évaluation des modèles afin de vérifier si l'ajout de l'information typographique et l'emploi des fichiers ALTO sont pertinents. Pour évaluer ces paramètres, chaque modèle a été entraîné sur un même jeu de données composé de 48 pages tirées des catalogues de la Société des artistes indépendants des années 1892, 1913 et 1923 pour les données d'entraînement et de 12 pages tirées de ces mêmes catalogues pour les données d'évaluation.

6.1 Entraîner un modèle dans *GROBID-dictionaries*

Pour entraîner un modèle dans *GROBID-dictionaries*, il est nécessaire de travailler à partir de Docker¹, un logiciel qui permet de lancer des applications dans des conteneurs. Ces derniers regroupent le code et les dépendances d'une application afin que celle-ci soit lancée rapidement dans n'importe quel environnement. Ainsi, Docker permet de faire fonctionner une application de manière uniforme sur tout système d'exploitation (Linux, Mac et Windows)². Ensuite, il faut télécharger la dernière version de *GROBID-dictionaries* dans le conteneur avant de lancer les premières commandes qui créent des fichiers d'entraînement pour chacun des niveaux encodés par l'outil de *machine learning*. Comme expliqué au chapitre 3³, cinq fichiers sont générés automatiquement dans le dossier de travail dont un fichier XML-TEI à annoter. Cette étape est fondamentale dans le

1. Docker, <https://www.docker.com/> (visité le 14/09/2020).

2. *ibid.*, « What is a Container ? ».

3. Se référer à la partie 3.1.1 sur le fonctionnement général de *GROBID-dictionaries*.

processus d'entraînement d'un modèle s'appuyant sur l'intelligence artificielle, puisque la machine va apprendre à partir de l'annotation des fichiers. Pour réaliser cette étape, il faut ouvrir le fichier dans un éditeur XML, tel Oxygen, et encoder le texte à partir du mode « auteur ». Ce mode permet d'éditer un document XML-TEI d'une façon plus accessible pour les utilisateur·rice·s non familier·ère·s avec l'éditeur, grâce à l'emploi d'une interface graphique qui surligne chaque partie textuelle encodée dans une balise d'une couleur précise. Ainsi, chaque balise est assignée à une couleur dans un document CSS, ce qui permet de mieux visualiser le texte encodé, comme sur la figure 6.1 ci-dessous.



FIGURE 6.1 – Capture d'écran du mode « auteur » d’Oxygen, septembre 2020.

Une fois tous les niveaux annotés et encodés, vient l'étape de l'entraînement du modèle à partir de ces données. Pour que l'entraînement soit efficace, il est essentiel de préparer des données d'entraînement et des données d'évaluation différentes, selon un rapport 80/20, c'est-à-dire comprenant environ 80% de données d'entraînement et 20% de données d'évaluation. Lors de chaque entraînement, la machine apprend à partir des données d'entraînement puis nous vérifions la performance du modèle en le comparant aux données d'évaluation. Pour chaque niveau, des résultats sont produits sous forme de pourcentages. Désormais, seul les *Field-level results*, c'est-à-dire les résultats de l'étiquetage de l'intégralité des séquences, sont précisés⁴. Autrement dit, le logiciel évalue le bon placement des balises XML-TEI⁵.

Pour chaque balise, sont précisées les informations suivantes :

4. Dans son mémoire, Lucie Rondeau du Noyer indique que *GROBID-dictionaries* donnait également les *Token-level results*, c'est-à-dire les résultats de l'étiquetage au niveau des *tokens*. Voir L. Rondeau du Noyer, *Encoder automatiquement des catalogues en XML-TEI...*, p. 40-41.

5. *ibid.*, p. 41.

| ===== Field-level results ===== | | | | | |
|------------------------------------|----------|-----------|--------|-------|---------|
| label | accuracy | precision | recall | f1 | support |
| <lemma> | 97.07 | 88.89 | 88.89 | 88.89 | 63 |
| <pc> | 95.4 | 94.36 | 94.36 | 94.36 | 195 |
| <sense> | 97.07 | 96.41 | 96.41 | 96.41 | 195 |
| all (micro avg.) | 96.51 | 94.48 | 94.48 | 94.48 | 453 |
| all (macro avg.) | 96.51 | 93.22 | 93.22 | 93.22 | 453 |
| ===== Instance-level results ===== | | | | | |
| Total expected instances: | 64 | | | | |
| Correct instances: | 52 | | | | |
| Instance-level recall: | | 81.25 | | | |

FIGURE 6.2 – Capture d'écran d'un résultat d'entraînement du niveau *lexical entry* dans *GROBID-dictionaries*, septembre 2020.

- **accuracy** vérifie la proportion de balises correctement placées.
- **precision** détermine le taux de balises bien placées qui ont été classées correctement.
- **recall** correspond au taux de balises correctement classées (bien placées ou non).
- **f1** est la moyenne harmonique entre le rappel et la précision.
- **support** indique le nombre de balises dans le document.

Le score **f1** est l'indicateur qui résume le mieux l'évaluation du modèle, toutefois il est essentiel de regarder également les scores de la précision et du rappel pour mieux comprendre le comportement de l'apprentissage supervisé. Ensuite, si les résultats de l'entraînement sont bons, c'est-à-dire supérieurs à 90%, l'interface graphique de *GROBID-dictionaries* peut être lancée.

6.2 Les modèles entraînés

6.2.1 Modèles entraînés à partir des PDF

Afin de vérifier l'efficacité de l'ajout de l'information typographique dans les *features* de *GROBID-dictionaries*, nous avons entraîné deux modèles à partir des PDF créés grâce à XSL-FO : le premier jeu de données contient l'information typographique tandis que le second en est dépourvu.

La table 6.1 figurant sur la page suivante présente les résultats de l'évaluation des cinq niveaux de *GROBID-dictionaries* obtenus à partir du jeu de données **trainingData_INDEPENDANTS -PDF-TYPO** composé de PDF dans lesquels l'information typographique est stockée.

| Balises | % accuracy precision recall f1 support | | | | |
|-------------------------------------|---|-------|-------|-------|-------|
| | | | | | |
| <i>dictionary segmentation</i> | | | | | |
| | <body> | 94.12 | 91.67 | 91.67 | 91.67 |
| | <footnote> | 100 | 100 | 100 | 100 |
| <i>dictionary body segmentation</i> | <headnote> | 97.06 | 100 | 88.89 | 94.12 |
| | <entry> | 98.43 | 98.44 | 98.44 | 98.44 |
| | <pc> | 99.21 | 100 | 98.39 | 99.19 |
| <i>lexical entry</i> | | | | | |
| | <lemma> | 97.07 | 88.89 | 88.89 | 88.89 |
| | <pc> | 95.4 | 94.36 | 94.36 | 94.36 |
| | <sense> | 97.07 | 96.41 | 96.41 | 96.41 |
| <i>form</i> | | | | | |
| | <desc> | 100 | 100 | 100 | 100 |
| | <name> | 98.95 | 98.39 | 98.39 | 98.39 |
| | <pc> | 98.95 | 98.46 | 98.46 | 98.46 |
| <i>sense</i> | | | | | |
| | <note> | 96.53 | 87.5 | 88.51 | 88 |
| | <num> | 100 | 100 | 100 | 100 |
| | <pc> | 96.53 | 88.3 | 89.25 | 88.77 |
| | <subSense> | 95.7 | 93.33 | 93.33 | 93.33 |

TABLE 6.1 – Résultats de l'évaluation de l'entraînement du modèle `trainingData_INDEPENDANTS-PDF-TYPO`.

La table 6.2 figurant ci-dessous présente les résultats de l'évaluation des cinq niveaux de *GROBID-dictionaries* obtenus à partir du jeu de données `trainingData_INDEPENDANTS-PDF-ST` composé du même corpus que le jeu de données précédent. Toutefois, celui-ci ne contient aucune information typographique.

Les résultats obtenus démontrent que l'ajout de l'information typographique dans le PDF a un réel impact. En effet, les résultats du modèle `trainingData_INDEPENDANTS-PDF-TYPO` sont tous supérieurs – excepté ceux de la balise `<lemma>` au niveau de la *lexical entry* – à ceux du modèle `trainingData_INDEPENDANTS-PDF-ST`. L'hypothèse de Mohamed Khemakhem, Lucie Rondeau du Noyer et Simon Gabay, sur l'amélioration du modèle grâce à l'ajout de nouvelles *features*⁶, est donc confirmée par ces résultats.

6. Simon Gabay, Lucie Rondeau du Noyer, Matthias Gille Levenson, Ljudmila Petkovic et Alexandre Bartz, « Quantifying the Unknown : How many manuscripts of the marquise de Sévigné still exist ? », dans *Digital Humanities DH2020*, Ottawa, 2020 (DH2020 Book of Abstracts), <https://hal.archives-ouvertes.fr/hal-02898929> (visité le 04/08/2020).

| Balises | % accuracy precision recall f1 support | | | | |
|-------------------------------------|---|-----------|--------|-------|---------|
| | accuracy | precision | recall | f1 | support |
| <i>dictionary segmentation</i> | | | | | |
| <body> | 94.12 | 91.67 | 91.67 | 91.67 | 12 |
| <footnote> | 100 | 100 | 100 | 100 | 12 |
| <i>dictionary body segmentation</i> | 97.06 | 100 | 88.89 | 94.12 | 9 |
| <entry> | 94.57 | 95.24 | 93.75 | 94.49 | 64 |
| <pc> | 97.67 | 100 | 95.16 | 97.52 | 62 |
| <i>lexical entry</i> | | | | | |
| <lemma> | 97.47 | 92.31 | 88.89 | 90.57 | 54 |
| <pc> | 95.09 | 97.35 | 90.74 | 93.93 | 162 |
| <sense> | 95.09 | 96 | 91.72 | 93.81 | 157 |
| <i>form</i> | | | | | |
| <desc> | 97.12 | 95.56 | 95.56 | 95.56 | 45 |
| <name> | 97.12 | 95.45 | 95.45 | 95.45 | 44 |
| <pc> | 97.84 | 97.73 | 95.56 | 96.63 | 45 |
| <i>sense</i> | | | | | |
| <note> | 95.56 | 85.06 | 86.05 | 85.55 | 86 |
| <num> | 99.47 | 100 | 98.26 | 99.12 | 172 |
| <pc> | 95.56 | 85.87 | 86.81 | 86.34 | 91 |
| <subSense> | 93.96 | 90.59 | 89.53 | 90.06 | 172 |

TABLE 6.2 – Résultats de l'évaluation de l'entraînement du modèle `trainingData_INDEPENDANTS-PDF-ST`.

6.2.2 Modèle entraîné à partir des ALTO

Au cours de l'été 2020, Mohammed Khemakhem a livré une version de *GROBID-dictionaries* capable de lire les fichiers ALTO. Nous avons donc entraîné un modèle, à partir du corpus présenté précédemment, pour comparer les résultats obtenus avec les PDF et les fichiers ALTO. L'information typographique ayant un réel impact sur les résultats, nous avons décidé d'entraîner uniquement un modèle à partir de fichiers ALTO dans lesquels les variances typographiques du texte sont incluses. Toutefois, Ljudmila Petkovic a mené un travail sur les fichiers ALTO afin de vérifier l'impact de la conservation de l'information typographique dans ces fichiers en comparant deux modèles entraînés avec et sans information typographique.

La table 6.3 présente les résultats des cinq niveaux de *GROBID-dictionaries* obtenus à partir du jeu de données `trainingData_INDEPENDANTS-ALTO` composé du même corpus tiré de catalogues de la Société des artistes indépendants.

Si au premier abord les résultats sont encourageants, le taux du bon placement des balises `<body>` et `<headnote>` au niveau de la *dictionary segmentation* étant de 100%, ces derniers se révèlent être moins bons que ceux obtenus avec les PDF contenant l'information typographique. Au niveau de la *dictionary body segmentation*, les taux sont inférieurs

| Balises | % accuracy precision recall f1 support | | | | |
|-------------------------------------|---|-------|-------|-------|-----|
| | | | | | |
| <i>dictionary segmentation</i> | | | | | |
| <body> | 100 | 100 | 100 | 100 | 9 |
| <headnote> | 100 | 100 | 100 | 100 | 9 |
| <i>dictionary body segmentation</i> | | | | | |
| <entry> | 82.76 | 77.61 | 83.87 | 80.62 | 62 |
| <pc> | 91.03 | 87.3 | 91.67 | 89.43 | 60 |
| <i>lexical entry</i> | | | | | |
| <lemma> | 98 | 90.38 | 95.92 | 93.07 | 49 |
| <pc> | 97.71 | 98.55 | 95.77 | 97.14 | 142 |
| <sense> | 95.43 | 95.77 | 93.15 | 94.44 | 146 |
| <i>form</i> | | | | | |
| <desc> | 96.45 | 95.56 | 93.48 | 94.51 | 46 |
| <name> | 97.16 | 97.67 | 93.33 | 95.45 | 45 |
| <pc> | 97.87 | 97.78 | 95.65 | 96.7 | 46 |
| <i>sense</i> | | | | | |
| <note> | 93.82 | 85.19 | 80.23 | 82.63 | 86 |
| <num> | 100 | 100 | 100 | 100 | 128 |
| <pc> | 94.03 | 86.25 | 80.23 | 83.13 | 86 |
| <subSense> | 92.54 | 86.72 | 86.05 | 86.38 | 129 |

TABLE 6.3 – Résultats de l'évaluation de l'entraînement du modèle `trainingData_INDEPENDANTS-ALTO`.

d'environ 15 points, et démontrent que le modèle a du mal à reconnaître les entrées des catalogues. Pour les trois autres niveaux, les taux sont tous légèrement inférieurs à ceux obtenus à partir des PDF.

Malgré ces résultats moins performants, ceux obtenus par Ljudmila Petkovic, à partir d'un corpus plus restreint de catalogues de vente de manuscrits⁷ sont encourageants. En effet, à l'exception des taux des balises `<lemma>` et `<sense>`, tous les autres sont supérieurs à 90%, taux indicateur du bon apprentissage de la machine, comme le montrent les résultats de la table 6.4. Par ailleurs, elle a également démontré que l'ajout de l'information typographique dans les fichiers ALTO rend le modèle plus performant⁸. Suite aux résultats obtenus par Ljudmila Petkovic, nous avons tenté de réduire le corpus pour entraîner le modèle à partir d'un nombre moins important de données, mais ce changement n'a pas amélioré de façon majeure les résultats. Nous avons donc émis l'hypothèse que certains niveaux du modèle entraîné à partir de fichiers ALTO nécessitent plus de données d'entraînement. Celle-ci a rapidement pu être vérifiée grâce à l'ajout de 8 pages annotées au niveau de la *dictionary body segmentation*, avec pour résultat une augmenta-

7. Ljudmila Petkovic a entraîné son modèle à partir d'un corpus constitué de 4 pages pour les données d'entraînement et de 1 page pour les données d'évaluation tirées d'un *Catalogue de lettres autographes et manuscrits* édité par Auguste Laverdet et d'un exemplaire de *L'Amateur d'autographes* édité par Jacques Charavay. Voir S. Gabay et L. Petkovic, *19th fixed-price and auction catalogues...*

8. *ibid.*

| Balises | % accuracy precision recall f1 support | | | | |
|-------------------------------------|---|-----------|--------|-------|---------|
| | accuracy | precision | recall | f1 | support |
| <i>dictionary segmentation</i> | 100 | 100 | 100 | 100 | 4 |
| | 100 | 100 | 100 | 100 | 4 |
| <i>dictionary body segmentation</i> | 97.94 | 0 | 0 | 0 | 1 |
| | 96.91 | 96.3 | 98.11 | 97.2 | 53 |
| <i><pc></i> | 96.91 | 94.87 | 97.37 | 96.1 | 38 |
| | | | | | |
| <i>lexical entry</i> | 99.66 | 0 | 0 | 0 | 1 |
| | 93.54 | 82.69 | 81.13 | 81.9 | 53 |
| <i><note></i> | 99.66 | 0 | 0 | 0 | 1 |
| | 99.66 | 98.11 | 100 | 99.05 | 52 |
| <i><pc></i> | 94.9 | 91.74 | 94.34 | 93.02 | 106 |
| | 93.88 | 83.02 | 83.02 | 83.02 | 53 |
| <i>form</i> | 95.65 | 90.91 | 95.24 | 93.02 | 42 |
| | 96.38 | 95.35 | 93.18 | 94.25 | 44 |
| <i><pc></i> | 96.38 | 91.11 | 97.62 | 94.25 | 42 |
| | | | | | |
| <i>sense</i> | 93.82 | 85.19 | 80.23 | 82.63 | 86 |
| | 98.81 | 0 | 0 | 0 | 1 |
| <i><dictScrap></i> | 97.62 | 100 | 83.33 | 90.91 | 12 |
| | 95.24 | 100 | 71.43 | 83.33 | 14 |
| <i><subSense></i> | 89.29 | 90.57 | 92.31 | 91.43 | 52 |

TABLE 6.4 – Résultats de l'évaluation de l'entraînement du modèle `trainingData_LAC_LAV_typo`

tion d'environ 8 points du taux de la balise `<entry>`. De ce fait, le modèle ALTO requiert un nombre plus important de données d'entraînement pour être efficace.

6.2.3 Comparaison des modèles : quelle efficacité ?

Les résultats observés précédemment sur les modèles entraînés à partir d'un corpus de catalogues d'exposition démontrent que l'ajout de l'information typographique, tant dans les fichiers ALTO que PDF, a un réel impact sur l'amélioration du modèle. L'intégration de nouvelles *features* précisant la position du texte sur la page et s'il est en gras, en italique ou dépourvu de toute information typographique supplémentaire a bien été prise en compte dans les processus de l'apprentissage machine et permet d'obtenir des modèles plus performants.

D'autre part, la modification des missions de stage au cours du mois de juillet a permis de démontrer que les modèles entraînés à partir de PDF contenant l'information typographique sont plus efficaces que ceux entraînés à partir de fichiers ALTO. En effet,

ces derniers nécessitent plus de données d’entraînement. Cela peut être dû au fait que *GROBID-dictionaries* a été développé pour lire et transformer des fichiers PDF et non des fichiers ALTO. Par ailleurs, ces derniers ont une structure plus complexe dans laquelle sont stockées plus d’informations. Pour rappel, les fichiers PDF créés à partir de XSL-FO ne rendent pas compte de la position exacte des mots sur la page ni de la taille de la police. Néanmoins, il serait sûrement possible d’y arriver en améliorant le code de la feuille de style XSL qui permet de transformer un fichier ALTO en fichier FO, avant sa conversion en PDF.

En outre, ces résultats remettent en cause l’utilité de créer des modèles à partir de fichiers ALTO. En effet, pour Béatrice Joyeux-Prunel et les besoins du projet Artl@s, il n’est pas utile de multiplier les formats de modèle, au risque de perdre du temps et de possiblement rendre le *workflow* plus confus. Nous avons donc décidé de conserver uniquement les modèles entraînés à partir de PDF⁹.

9. Les autres modèles restent toutefois disponibles en ligne sur le dépôt GitHub que j’ai créé à l’occasion du stage : <https://github.com/carolinecorbieres/ArtlasCatalogues>

Troisième partie

Pérenniser le projet : proposition
d'un *workflow* réutilisable

Chapitre 7

Les étapes du *workflow*

Le principal objectif de ce stage a été de créer un *workflow* réutilisable par les membres du projet d'Artl@s pour encoder automatiquement les catalogues d'exposition. Celui-ci doit permettre de traiter un catalogue depuis son format de récupération (PDF ou JPEG) jusqu'à son encodage en XML-TEI. Ce travail concerne uniquement la partie textuelle des catalogues. Leurs métadonnées, entrées dans un fichier csv rempli au fur et à mesure par des membres du projet¹, sont récupérées grâce à une feuille de style XSL créée par Auriane Quoix qui génère automatiquement les *headers* des catalogues. Cette étape a été ajoutée au *workflow*² de façon à mutualiser notre travail, qui est également regroupé au sein d'un même dossier sur le GitLab de Béatrice Joyeux-Prunel³. Outre cette étape supplémentaire, qui permet de compléter les fichiers XML-TEI, le *workflow* a été pensé en six étapes :

- l'import du catalogue en format JPEG ou PDF ;
- l'OCRisation du catalogue et l'exportation de la transcription dans des fichiers ALTO ;
- la transformation des fichiers ALTO en PDF ;
- l'encodage automatique du catalogue avec *GROBID-dictionaries* ;
- la transformation de l'encodage en sortie de *GROBID-dictionaries* et la réunion du <header> et du <body> dans un même fichier .xml ;
- la transformation du fichier XML-TEI en CSV.

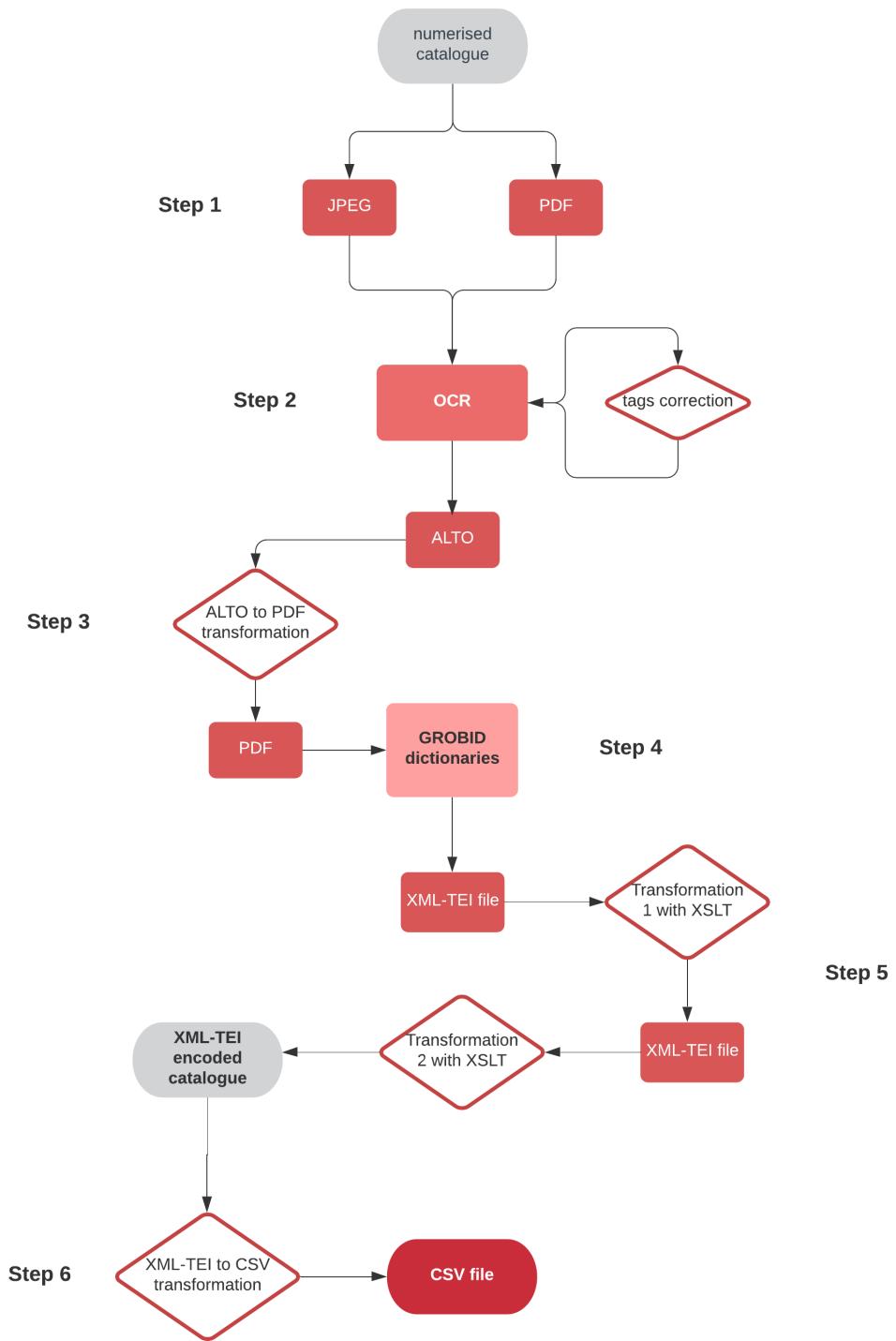
7.1 Importer les catalogues

L'import des catalogues est une étape primordiale pour leur OCRisation. Nous devons récupérer les images des pages des catalogues pour pouvoir ensuite les importer dans

1. Ce fichier csv a été rempli en majeure partie par Barbara Topalov.

2. Voir Annexe B.1 Step 0.

3. Béatrice Joyeux-Prunel, *Visual Contagions*, GitLab, <https://gitlab.unige.ch/Beatrice.Joyeux-Prunel/visual-contagions/-/tree/master/Catalogues> (visité le 10/09/2020).

FIGURE 7.1 – Schéma récapitulant les 6 étapes du *workflow*.

un logiciel d’OCR. Ces images doivent être de bonne qualité afin que la transcription de l’OCR soit la plus correcte possible. En effet, des documents mal numérisés génèrent plus d’erreurs dans l’OCR. Frédéric Clavert rajoute qu’un texte mal reconnu par l’OCR engendre par la suite une perte d’information : par exemple, les mots mal transcrits peuvent fausser les résultats d’une étude linguistique⁴.

L’une des solutions proposées pour récupérer les images des catalogues est d’utiliser un script python qui importe les images à partir de l’API (*Application Programming Interface*) IIIF (*International Image Interoperability Framework*). Ce script s’appuie sur l’URL de l’API Image, qui peut être reconstituée à l’aide de commandes. Par défaut, le script reconstitue l’URL d’une API Image de Gallica, l’une des principales sources de catalogues d’exposition numérisés. Il est toutefois possible de modifier l’URL grâce à différents paramètres. Chacun d’entre eux correspond à une partie de la structure d’une URL : le protocole (`-s`), le nom de domaine (`-d`), le préfixe (`-p`), l’identifiant du document⁵ et le nom du manifeste dans l’URL (`-m`). Une fois l’URL de l’API Image reconstituée, le script accède aux images et les télécharge toutes. Celles-ci sont stockées dans la balise json nommée `canvases`. L’API Image de IIIF permet notamment de récupérer des images de haute qualité.

Étant conscient·e·s que tous les catalogues numérisés ne sont pas accessibles via une API IIIF, nous proposons également aux utilisateur·rice·s du *workflow* de récupérer la version PDF d’un catalogue. Celle-ci peut-être convertie en images par Transkribus.

Enfin, si le catalogue à traiter n’a pas de version numérique, nous conseillons à l’utilisateur·rice de récupérer les pages du catalogue de préférence en format JPEG ou PDF.

7.2 OCR et création de fichiers ALTO

Ensuite l’utilisateur·rice doit OCRiser le catalogue afin d’en récupérer la transcription dans laquelle est contenue l’information typographique. Nous lui suggérons de réaliser cette étape avec Transkribus, néanmoins, il se peut qu’à l’avenir d’autres logiciels d’OCR soient proposés. Le principal avantage de Transkribus est qu’il est possible d’y partager un projet avec des personnes choisies, pour qu’ils ou elles puissent y contribuer. De plus, c’est avec ce logiciel que travaillent depuis quelques années les membres des équipes de *GROBID-dictionaries* et d’e-ditiones⁶. D’autre part, son interface est plus avenante que d’autres logiciels d’OCR comme Kraken qui s’utilise avec le terminal.

Une fois le catalogue d’exposition transcrit, l’utilisateur·rice doit l’exporter au for-

4. Frédéric Clavert, « Une histoire par les données ? Le futur très proche de l’histoire des relations internationales », *Bulletin de l’Institut Pierre Renouvin*, N° 44–2 (1er nov.2016), <https://www.cairn.info/revue-bulletin-de-l-institut-pierre-renouvin-2016-2-page-119.htm> (visité le 08/08/2020).

5. Cet identifiant prend la forme d’un chemin, cela correspond à l’identifiant ark à la BnF.

6. L. Rondeau du Noyer, *Encoder automatiquement des catalogues en XML-TEI...*, p. 31.

mat .txt et lancer un premier script python qui permet de pointer les erreurs de balisage du gras et de l’italique⁷. Il ou elle est invitée à corriger un certain type d’erreurs dans l’espace de transcription de Transkribus, les autres étant ensuite automatiquement corrigées par un second script. Enfin, il ou elle exporte la transcription partiellement corrigée au format ALTO avant de lancer le second script qui permet de corriger les erreurs de balisage restantes et de transformer les fichiers ALTO pour qu’ils soient conformes au format accepté par *GROBID-dictionaries*. Pour lancer les deux scripts, il est indispensable de passer par le terminal.

7.3 De l’ALTO au PDF en passant par XSL-FO

Nous offrons ensuite la possibilité à l’utilisateur·rice de transformer les fichiers ALTO en PDF. Cette étape a été rajoutée au cours du mois de juillet 2020 en attendant que *GROBID-dictionaries* puisse lire les fichiers ALTO. Malgré le fait que depuis août 2020 le logiciel soit capable de les lire, nous avons conservé cette étape car les résultats de l’entraînement de modèles à partir de fichiers PDF démontrent qu’ils sont plus efficaces que les modèles entraînés à partir de fichiers ALTO.

La transformation des fichiers ALTO en PDF se fait en deux étapes : les fichiers sont premièrement convertis en fichiers XSL-FO grâce à une feuille de style XSLT. Puis ils permettent de générer des fichiers PDF à l’aide d’un processeur. Comme expliqué précédemment, nous avons choisi d’utiliser le processeur Render X⁸. Toutes les étapes liées à son installation dans l’éditeur XML Oxygen sont détaillées dans le *workflow*⁹. L’avantage d’XSL-FO est qu’il nous a permis de conserver l’information typographique dans le PDF, contrairement à ceux produits en sortie d’un logiciel d’OCR, qui en sont dépourvus. Ces deux étapes de transformation sont à réaliser dans l’éditeur XML Oxygen au sein duquel a été créé un projet qui regroupe tous les dossiers du *workflow*. Un projet Oxygen permet de regrouper différents fichiers et d’y accéder facilement. Il peut par ailleurs être partagé et offre également la possibilité d’appliquer un scénario de transformation, tel une feuille de style XSL, à tous les fichiers d’un dossier. De cette façon, l’ensemble des fichiers ALTO – chacun décrivant une page d’un catalogue – peut être traité en une seule fois.

Enfin, les pages recensant des entrées de catalogue doivent être rassemblées entre elles sous la forme d’un unique document PDF. Pour cela, nous utilisons l’outil cpdf¹⁰ qui permet de réaliser différentes opérations sur des documents PDF via le terminal. Il est par exemple possible de regrouper une sélection de pages ou toutes celles contenues dans un dossier, mais aussi de séparer toutes les pages d’un PDF. Cet outil permet d’éviter une

7. Ces étapes sont développées plus en détail au chapitre 4.

8. Voir au chapitre 5 la partie dédiée à XSL-FO, p.50.

9. Voir Annexe B.4 Step 3.

10. *Coherent PDF Command Line Tools Community Release*, <https://community.coherentpdf.com/> (visité le 11/09/2020).

perte de l'information que peuvent engendrer des logiciels de manipulation de documents PDF en ligne.

7.4 Encoder automatiquement des catalogues grâce à *GROBID-dictionaries*

Une fois le catalogue prêt, c'est-à-dire quand toutes les pages sont assemblées dans un seul document PDF, l'utilisateur·rice est invité·e à choisir le modèle d'entraînement correspondant le mieux au type d'entrées du catalogue qu'il ou elle souhaite encoder. À ce jour, deux modèles ont été entraînés : un à partir des catalogues d'exposition de la Société des artistes indépendants et un second à partir des catalogues de biennales. Chacun de ces deux modèles a été entraîné avec des PDF contenant l'information typographique pour offrir aux membres d'Artl@s des modèles plus performants. Après avoir synchronisé le dossier correspondant au modèle sélectionné, il ou elle doit lancer les entraînements des cinq niveaux du logiciel de *machine learning* dans le terminal. Ensuite, il ou elle peut accéder au serveur local, alors synchronisé au jeu de données entraîné précédemment. De là, il ou elle charge le catalogue puis *GROBID-dictionaries* se charge de l'encoder en XML-TEI. Le serveur propose ensuite de télécharger le résultat.

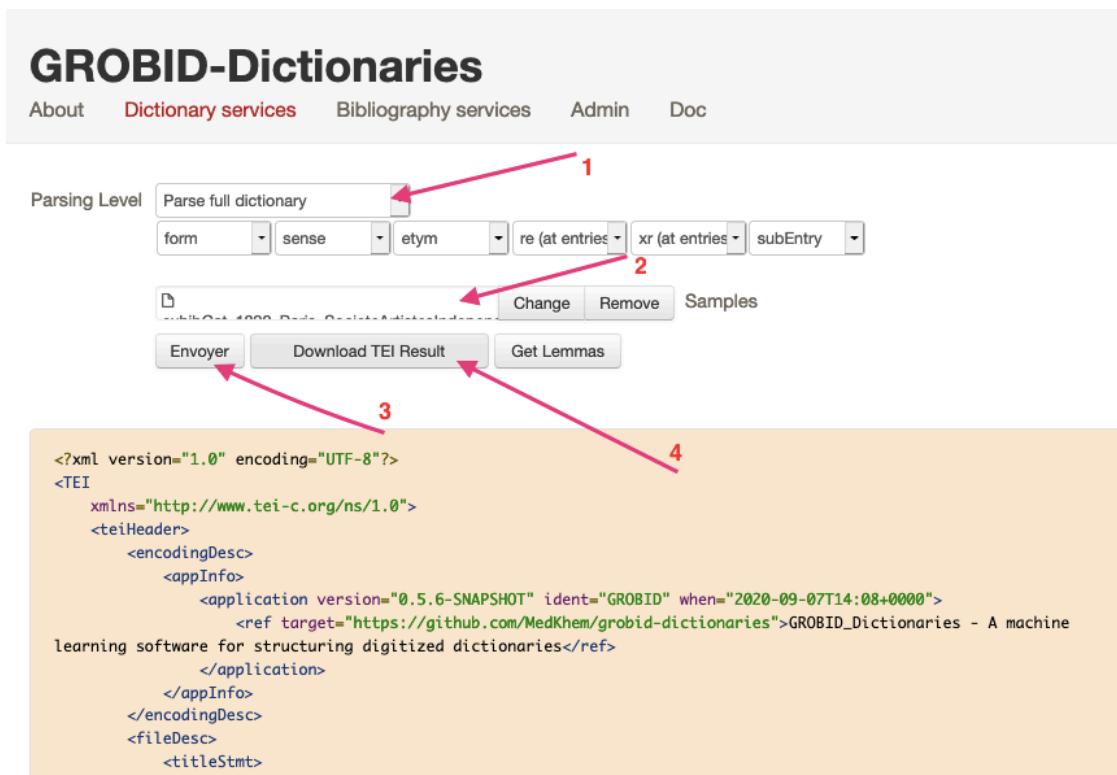


FIGURE 7.2 – Capture d'écran du serveur local de *GROBID-dictionaries*, septembre 2020.

Outre les démarches pour encoder automatiquement les catalogues, nous proposons également aux utilisateur·rice·s un guide pour qu'ils ou elles puissent ajouter de nou-

velles données d’entraînement à un modèle existant ou en créer un nouveau. Pour cela, il faut entrer des commandes spécifiques dans le terminal qui génèrent des données d’entraînement¹¹ dans le dossier dédié au modèle. Chaque niveau de *GROBID-dictionaries* doit contenir des données d’entraînement et des données d’évaluation annotées en XML-TEI, accompagnées de fichiers `.raw` contenant le texte brut et les *features*. Un modèle est considéré comme étant capable d’encoder efficacement des fichiers quand ses résultats d’évaluation sont supérieurs à 90% pour chaque niveau. Dès que ce pourcentage est atteint, le modèle peut être utilisé.

7.5 Améliorer l’encodage en sortie de *GROBID-dictionaries* avec XSLT

L’avant-dernière étape du *workflow* consiste à la transformation et la correction, en deux étapes, du catalogue encodé en XML-TEI par *GROBID-dictionaries*. Deux feuilles de style XSL et ODD, s’appuyant sur le précédent travail de Lucie Rondeau du Noyer, ont été créées afin de transformer le fichier récupéré en sortie de *GROBID-dictionaries* et d’en valider la structure. La première feuille de style renomme toutes les balises pour qu’elles correspondent à l’encodage choisi pour les catalogues d’exposition¹² et numérote les œuvres exposées. Cette deuxième opération récupère le numéro de l’œuvre et l’insère dans un attribut `@n` puis vérifie, à l’aide d’une règle Schematron¹³, l’ordre des numéros : ceux-ci doivent se suivre dans l’ordre numérique. Ainsi, chaque fois que deux numéros ne se suivent pas, l’erreur est pointée par l’éditeur XML Oxygen. Ces erreurs proviennent généralement d’une mauvaise reconnaissance des caractères par le logiciel d’OCR et sont facilement corrigibles à la main. D’autre part, cette feuille de style XSL associe une première ODD au fichier qui permet de le valider. Celle-ci vérifie notamment que chaque entrée est bien composée d’un `<desc>` et d’un ou plusieurs `<item>` et que chaque `<item>` comprend un `<num>` et un `<title>`.

Une fois la première transformation validée, une deuxième est appliquée au fichier. Tout comme la première, elle vérifie l’ordre de la numérotation des œuvres mais aussi des entrées et associe le fichier à l’ODD finale, co-écrite avec Auriane Quoix, qui permet de valider tous les documents encodés pour les projets Artl@s et Visual Contagions¹⁴. Cette ODD autorise certains enchaînements de balises et valide également les métadonnées encodées dans le `<header>`. De plus, la deuxième feuille de style XSL permet d’associer le `<header>`, encodé dans un autre fichier `.xml`, au corps de texte du catalogue et de donner

11. Voir l’annexe C.

12. Se référer à la partie 2.2.1 du chapitre 2.

13. Le langage Schematron permet de définir des contraintes qui rendent le document XML valide uniquement si ces règles sont suivies.

14. Dans le cadre du projet Visual Contagions, les métadonnées de périodiques sont encodées dans un document `.xml`.

à chaque entrée et œuvre un identifiant qui leur est propre. Dans les métadonnées sont encodées toutes les informations retrouvées dans BasArt sur la source et la description de l'exposition. Y sont également renseignés les logiciels utilisés pour l'encodage des entrées (Trankribus et *GROBID-dictionaries*) ou encore les personnes qui ont participé à la production des données encodées.

7.6 Du format XML-TEI au CSV : transformation des fichiers grâce à XSLT

Enfin, la dernière étape du *workflow* consiste à transformer le fichier XML-TEI obtenu précédemment en fichier CSV. Ce dernier format est celui utilisé jusqu'alors par les membres d'Artl@s pour téléverser les données du catalogue dans BasArt. Auparavant ils ou elles remplissaient manuellement les lignes du document CSV à partir des données contenues dans la liste des œuvres exposées. Désormais, il est possible de récupérer les informations encodées dans le fichier XML-TEI par *GROBID-dictionaries* et de les placer dans les colonnes correspondantes du CSV.

Pour cela, nous avons édité une feuille de style XSL qui crée un fichier CSV et remplit les colonnes correspondant aux balises des informations que nous souhaitons récupérer. Ainsi, pour chaque œuvre exposée est créée une ligne dans laquelle les informations propres à l'artiste sont dispersées entre deux colonnes et celles concernant l'œuvre entre trois colonnes. Au total, le fichier est composé d'une soixante-dizaine de colonnes correspondant aux différentes entrées de la base de données d'Artl@s. Pour l'instant seules les colonnes `name`, `trait`, `num`, `title` et `desc` sont remplies automatiquement. Les autres seront complétées manuellement par des membres de l'équipe en fonction des informations récupérées avec *GROBID-dictionaries*. Cette étape sera également l'occasion pour l'équipe de relire les données et de les corriger si besoin avant leur envoi dans la base de données.

Pour le moment, le *workflow* permet uniquement d'encoder les catalogues d'exposition jusqu'à un certain niveau, qui correspond à celui de *GROBID-dictionaries*. Néanmoins, une autre couche d'encodage pourrait être ajoutée afin d'affiner le code. Cela permettrait de récupérer des informations biographiques ou propres à la description des œuvres dans le but d'automatiser la récupération des données dans le CSV et d'alimenter BasArt. Pour extraire ces informations, plusieurs solutions existent : l'équipe pourrait utiliser un logiciel NER (*Named Entity Recognition*), tel *GROBID-ner*, qui recherche et classe des entités nommées dans des catégories prédéfinies (noms de personnes, de lieux, d'institution, quantités, monnaies...)¹⁵ ou travailler à partir de scripts qui extraient des

15. Léa Saint-Raymond a utilisé une technologie NER pour extraire des données issues d'un corpus de catalogues de ventes aux enchères de tableaux. Léa Saint-Raymond et Antoine Courtin, « Enriching and Cutting : How to Visualize Networks Thanks to Linked Open Data Platforms. », *Artl@s Bulletin*,

données en s'appuyant notamment sur des expressions régulières¹⁶.

6-3 (30 nov. 2017), p. 93.

16. Simon Gabay et Matthias Gille Lenvenson ont opté pour cette deuxième solution pour affiner l'encodage des catalogues de vente de manuscrits. S. Gabay et M. Gilles Levenson, *katabase/reconciliation...*

Chapitre 8

Un *workflow* « *user friendly* » ? Choix et limites

L'intérêt principal de la création de ce *workflow*, outre de conserver une trace du travail réalisé lors de ce stage, est d'assurer sa pérennité en faisant en sorte qu'il soit réutilisable par autrui. Il a en majeure partie été pensé pour que les membres du projet Artl@s puissent se l'approprier, tant à travers les applications proposées que par son format. Néanmoins, l'accessibilité de certaines des étapes peut être remise en cause du fait qu'elles requièrent des connaissances en informatique.

8.1 Forme et utilisateur·rice·s du *workflow*

8.1.1 Qui sont les utilisateur·rice·s du *workflow* ?

Les principales personnes amenées à travailler à partir du *workflow* sont les membres de l'équipe d'Artl@s et ceux des équipes partenaires du projet.

L'équipe d'Artl@s est principalement composée de chercheur·se·s en histoire de l'art, géographie, sociologie et sciences de l'information¹. Certain·ne·s d'entre elles et eux se sont formés aux humanités numériques, notamment dans les technologies de base de données PostGIS pour alimenter BasArt et créer d'autres bases de données géoréférencées comme GeoMAP². Même s'ils ou elles se sont spécialisé·e·s dans les bases de données, ils ou elles ne maîtrisent pas forcément les lignes de commande, le langage XML-TEI ou encore les outils de *machine-learning*. D'autres encore portent un intérêt particulier à la culture des humanités numériques, sans forcément en savoir manipuler les outils.

De même, les membres des équipes de recherche partenaires du projet travaillant dans différentes universités, comme celle de Tokyo, celle de São Paulo ou encore l'Univers-

1. « L'équipe principale », Artl@s...

2. *ibid.*

sité Purdue³, n'ont pas tou·te·s été formé·e·s aux humanités numériques.

Pour pallier à ce manque, quelques formations ont été mises en place par les membres d'Artl@s. Toutefois, il faut prendre en compte le fait que certaines étapes, nécessitant des connaissances plus poussées en informatique ne pourront pas être réalisées par tous les membres du projet. Néanmoins, pour que chacun·ne puisse participer à BasArt, certaines étapes ont été pensées de manière à n'exiger aucune connaissance poussée dans le numérique et sont donc adaptées aux néophytes.

8.1.2 Forme et applications sélectionnées

Afin que le *workflow* soit accessible à tous les membres de l'équipe et aux partenaires dispersé·e·s dans le monde entier, il a été décidé de le déposer sur le GitLab de Béatrice Joyeux-Prunel⁴, un logiciel de gestion de projets git *open source* qui permet de travailler de manière collaborative. L'un des avantages de GitLab est qu'il est possible d'assigner des rôles aux membres du projet, chaque rôle permettant d'accéder à des fonctionnalités différentes⁵. D'autre part, l'équipe étant internationale, les étapes du *workflow* ont toutes été rédigées en anglais, langue privilégiée des humanités numériques. Chacune d'entre elles, prenant la forme d'un tutoriel, est complétée par des captures d'écran qui rendent ainsi plus compréhensibles certaines sous étapes⁶. Elles ont également été rédigées de la manière la plus simple et claire possible de façon à privilégier leur accessibilité. Par exemple, pour les étapes requérant l'emploi du terminal, toutes les commandes ont été pré-remplies afin que l'utilisateur·rice ait simplement à compléter une partie du chemin⁷. À titre d'illustration, pour la commande suivante « `cd YOUR_PATH_TO_THE_FOLDER/visual-contagions/Catalogues/1_ImportCatalogues` », il ou elle doit remplacer « `YOUR_PATH_TO_THE_FOLDER` » par le chemin de son ordinateur menant au dossier nommé « `visual-contagions` », qui correspond au dépôt git téléchargé.

Afin de rendre le *workflow* le plus accessible possible, nous nous sommes tourné·e·s vers des applications dont l'interface graphique est plus avenante et intuitive que le terminal. Outre *GROBID-dictionaries*, les deux autres applications utilisées dans le *workflow* sont Transkribus et Oxygen.

3. « L'équipe principale », *Artl@s...*

4. Il est disponible à l'adresse suivante : <https://gitlab.unige.ch/Beatrice.Joyeux-Prunel/visual-contagions/-/tree/master/Catalogues>.

5. « Permissions », *Gitlab*, <https://docs.gitlab.com/ee/user/permissions.html> (visité le 22/09/2020).

6. Voir l'annexe B.

7. Les parties à compléter sont spécifiées en majuscules.

Transkribus

Transkribus est un logiciel de reconnaissance optique de caractères qui permet de transcrire des documents historiques, dont des manuscrits. Il a été pensé par la coopérative européenne READ pour les historien·ne·s et les archivistes dans le but d'automatiser la transcription des documents⁸. L'application ayant été créée pour des personnes issues des sciences humaines et sociales, ses développeur·se·s ont conçu une interface graphique plutôt accessible. De même, l'équipe de READ a réalisé des tutoriels disponibles en ligne pour s'autoformer et prendre en main l'application⁹. Les membres de l'équipe amenés à réaliser l'OCRisation des catalogues sont donc invité·e·s à s'autoformer ou à suivre une formation de prise en main du logiciel. D'autant plus que, pour les besoins du *workflow*, seule la maîtrise des opérations de base est requise. L'utilisateur·rice doit uniquement savoir charger des documents dans Transkribus, lancer l'OCR et l'HTR, opérer des corrections dans la transcription et exporter celle-ci au bon format. Il ou elle n'a nul besoin de savoir maîtriser les fonctionnalités avancées du logiciel d'OCR à savoir la création de modèles et leur entraînement. Toutefois, l'équipe READ a également créé des tutoriels pour apprendre aux utilisateur·rice·s à entraîner des modèles HTR dans Transkribus.

Oxygen

Oxygen est un éditeur de développement XML qui permet d'éditer principalement des documents XML, des schémas validant des fichiers XML et des transformations XSLT convertissant les fichiers XML. Ce logiciel est notamment utilisé pour créer des documents XML-TEI valides. Dans le cadre du projet Artl@s, les utilisateur·rice·s sont invité·e·s à simplement transformer des documents grâce à des feuilles de style XSL pré-écrites et à corriger les éventuelles erreurs dans le document XML-TEI créé par *GROBID-dictionaries*. Toutes les manipulations nécessaires pour lancer les transformations XSLT sont précisées dans le *workflow* et illustrées de captures d'écran¹⁰. De cette façon, il ou elle n'a, à aucun moment, besoin d'éditer un document XML dans le logiciel. Si l'utilisateur·rice souhaite apprendre à maîtriser le logiciel, il ou elle peut se tourner vers les nombreuses vidéos explicatives mises à disposition de la communauté Oxygen par l'équipe du logiciel¹¹. Celles-ci prennent différentes formes (tutoriel, webinaire et conférence) et portent sur toutes les technologies développées pour le logiciel. Il est également possible de filtrer les vidéos par langage et ainsi se former à l'édition d'un certain type de document. Tout comme Transkribus, Oxygen est une application qui touche un public de chercheur·se·s en sciences humaines et sociales, initialement non

8. « Transkribus », *READ-COOP*, <https://readcoop.eu/transkribus/> (visité le 19/09/2020).

9. « Main Page », *Transkribus Wiki*, https://transkribus.eu/wiki/index.php/Main_Page (visité le 19/09/2020).

10. Voir l'annexe B.

11. « Vidéos », *Oxygen XML Editor*, <https://www.oxygenxml.com/videos.html> (visité le 20/09/20).

formé au numérique. L’interface graphique d’Oxygen s’appuie sur celle d’un éditeur de texte avec des onglets et des icônes situés en haut de l’écran, le texte encodé étant au centre. Ce logiciel permet également de travailler sur un dossier entier grâce à la création d’un projet. Il est ainsi possible d’accéder depuis l’application à toute l’arborescence du dossier dans lequel est placé le projet. Pour faciliter l’accès aux différents catalogues et à toutes les feuilles de style XSL créées pour le *workflow*, nous utilisons un projet nommé `ExhibitionCatalogues.xpr`. Placé à la racine du dossier dans lequel se trouve le *workflow*, il permet à tout·e utilisateur·rice d’ouvrir l’ensemble du dossier dans l’éditeur XML et de travailler directement dessus. Nous avons également privilégié l’utilisation de scripts XSLT à celle des scripts python, puisqu’ils sont lancés depuis Oxygen et non depuis le terminal, dont l’interface est moins avenante pour les néophytes.

Enfin, la dernière étape du *workflow* permet d’obtenir un fichier `.csv`, format connu des membres du projet Artl@s. Le retour vers ce type de fichier permet de faciliter à la fois la rentrée des données dans BasArt, ainsi que leur relecture par les utilisateur·rice·s qui doivent les déplacer dans les colonnes correspondantes. Ce choix, émis par Béatrice Joyeux-Prunel, a l’avantage de proposer un environnement familier aux personnes travaillant sur le projet, notamment aux étudiant·e·s de master qui contribuent à la rentrée des données dans BasArt.

8.2 Limites de l’accessibilité du *workflow*

Malgré le souhait de rendre le *workflow* le plus accessible possible, certaines étapes requièrent la maîtrise des lignes de commande dans le terminal. Même si toutes les commandes ont été précisées dans la chaîne de traitement, l’usage du terminal peut s’avérer être plus difficile pour les néophytes. Nous avons donc tenté de guider le plus possible les utilisateur·rice·s dans l’exécution des commandes ; néanmoins, il pourrait être utile d’ajouter au *workflow* un guide à destination des néophytes pour leur apprendre à utiliser le terminal et définir précisément l’action de chaque commande, à l’image du guide pour débutant présenté dans le rapport *The BL2D Mesh Generator : Beginner’s Guide, User’s and Programmer’s Manual* par Patrick Laug et Houman Borouchaki¹².

L’emploi du terminal est parfois inévitable : nous l’utilisons pour lancer des scripts python ou bash, lancer les entraînements dans *GROBID-dictionaries* et accéder au serveur local de l’application ainsi que pour mettre à jour le dépôt sur GitLab. Dans le cas de l’étape 1, qui consiste à récupérer le catalogue en format JPEG ou PDF, nous offrons la possibilité à l’utilisateur·rice d’éviter de passer par le terminal, même si nous en recommandons l’usage : il ou elle peut récupérer un PDF au lieu de télécharger les

12. Patrick Laug et Houman Borouochaki, *The BL2D Mesh Generator : Beginner’s Guide, User’s and Programmer’s Manual*, INRIA, 1996, p. 5-11.

images des pages du catalogues via un script python. Outre cette seconde option proposée, toutes les autres étapes requérant l'emploi du terminal n'ont, pour l'instant, aucune autre alternative. La correction de la transcription et la transformation des fichiers ALTO est obligatoirement lancée grâce à une commande qui exécute le script dans le terminal tout comme l'entraînement des modèles dans *GROBID-dictionaries*, qui est également lancé via des commandes. De plus, il est indispensable de maîtriser les commandes git pour déposer le travail réalisé en interne sur le GitLab du projet.

Le fait que la principale application du *workflow* requière la maîtrise des lignes de commande limite son accessibilité. Seul·e·s les membres du projet y étant formé·e·s ou souhaitant s'y former pourront prolonger le travail réalisé lors de ce stage, à savoir l'encodage automatique des catalogues. La volonté de faire une chaîne de traitement « *user friendly* » est donc en partie respectée puisque l'utilisateur·rice est accompagné·e dans chaque étape et que l'emploi de certaines applications et de scripts XSLT, plus accessibles, a été privilégié à celui de scripts python. Néanmoins, le *workflow* reste difficile d'accès aux néophytes pour lesquel·le·s il faut prévoir un accompagnement pouvant prendre la forme d'une formation. Une autre solution envisageable pour pallier au problème de l'accès à la chaîne de traitement serait de développer une interface qui éviterait de passer par le terminal. Toutefois, celle-ci nécessiterait des moyens humains et budgétaires probablement supérieurs à ceux demandés pour la formation des membres d'Artl@s. Par ailleurs, si GitLab n'a pas prévu de proposer une interface graphique permettant de déposer manuellement de nombreux documents, c'est qu'il est indispensable de passer par le terminal. De ce fait, même si nous souhaitons rendre les humanités numériques les plus accessibles possible, la formation des utilisateur·rice·s aux outils employés au sein d'un projet demeure inévitable.

Chapitre 9

Utilisations, perspectives et éventuelles améliorations du *workflow*

Cette dernière partie du mémoire vise à présenter un exemple d'utilisation des données récupérées du *workflow* ainsi que les premières perspectives de son emploi. Elle est aussi l'occasion de revenir sur le souhait de l'équipe de faire un *workflow* entièrement *open source* et de proposer d'éventuelles améliorations pour atteindre ce but.

9.1 Exemple d'utilisations du *workflow* et premières perspectives

9.1.1 Exemple d'utilisation

Dans le cadre de ce stage, j'ai été chargée d'encoder automatiquement les catalogues de la Société des artistes indépendants disponibles sur Gallica¹. Les données de ces documents, versées sur BasArt, peuvent être interrogées de diverses manières : par exemple, les membres d'Artl@s se sont intéressé·e·s à cartographier les expositions pour étudier la circulation des œuvres aussi que celle des artistes. Les multiples informations récoltées sur les artistes permettent également d'analyser leur origine ou encore leur genre. Nous allons nous intéresser à cette dernière donnée pour comparer la proportion d'artistes femmes qui ont exposé des œuvres au Salon des Indépendants de 1890, 1892 et 1923.

Vers la fin XIX^e siècle, le nombre de femmes artistes ne cesse d'augmenter par rapport au début du siècle². Néanmoins, elles demeurent sous-représentées dans les salons

1. Environ quarante catalogues allant des années 1890 aux années 1950 sont accessibles en ligne sur le site de Gallica.

2. Denise Noël, « Les femmes peintres dans la seconde moitié du XIX^e siècle », *Clio. Femmes, Genre, Histoire*, 19, 2004, <http://journals.openedition.org/clio/646>, (visité le 24/09/2020).

artistiques et éprouvent des difficultés à se démarquer face aux artistes masculins. En effet, la misogynie régnante amène de nombreux critiques à dévaloriser la peinture des femmes artistes, dont l'art est qualifié d'amateurisme³. L'interrogation des données des catalogues confirme l'idée que les femmes sont sous-représentées au Salon des Indépendants, néanmoins elle permet de quantifier le pourcentage de femmes qui ont exposées dans ce salon. Les trois figures suivantes (9.1, 9.2 et 9.3) montrent la proportion d'artistes femmes qui ont présenté des œuvres au salon de la Société des artistes indépendants (en bleu) : en 1890 elles représentent 17,65 % des exposant·e·s, deux ans plus tard, leur nombre diminue, en effet seulement 10,42 % des artistes sont des femmes. Toutefois, les données du salon de 1923 dévoilent que leur nombre est croissant puisqu'elles représentent désormais 22,66 % des exposant·e·s alors même que le nombre de participant·e·s a été multiplié par six en une trentaine d'années.

Genre des artistes à l'exposition de la Société des artistes indépendants de 1890

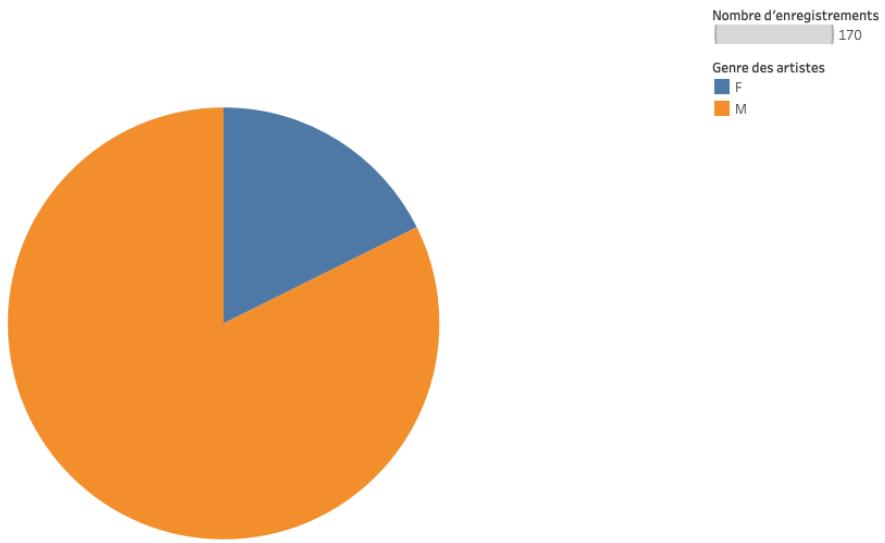


FIGURE 9.1 – Proportion des artistes femmes au salon de la Société des artistes indépendants de 1890.

Même si la présence des femmes est minoritaire dans les salons, elle tend à augmenter au fil du temps. Les artistes femmes exposent également dans des salons et expositions qui leurs sont exclusivement consacrés. Ces dernières expositions pâtissent de l'ignorance de la critique mais aussi d'articles incendiaires les décrivant comme insignifiantes ou manquant d'innovation⁴. Malgré cela, ces expositions non-mixtes ont permis aux artistes femmes d'exposer et d'être soutenues économiquement et socialement par des

3. Catherine Dossin et Hanna Alkema, « Women Artists Shows · Salons · Societies : Towards a Global History of All-Women Exhibitions », *Artl@s Bulletin*, 8-1, 2019, p. 7.

4. *ibid.*, p. 7.

Genre des artistes à l'exposition de la Société des artistes indépendants de 1892

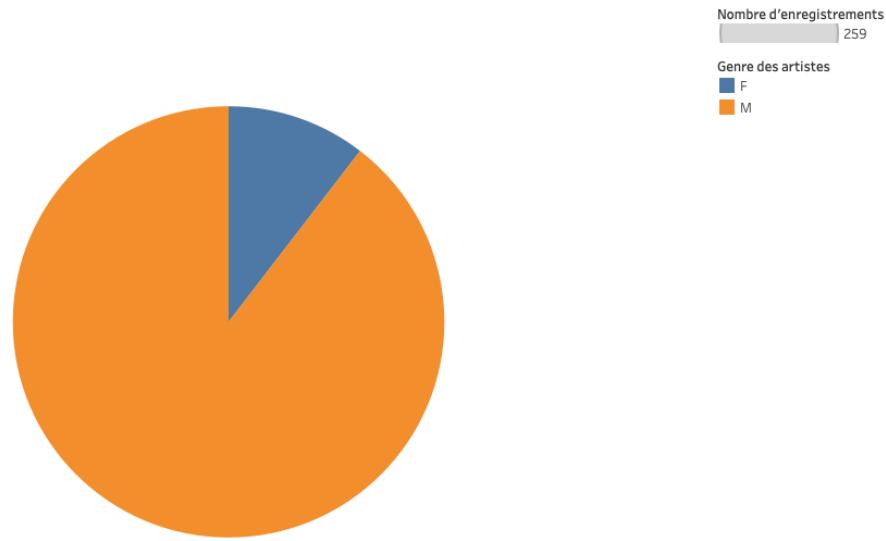


FIGURE 9.2 – Proportion des artistes femmes au salon de la Société des artistes indépendants de 1892.

Genre des artistes à l'exposition de la Société des artistes indépendants de 1923

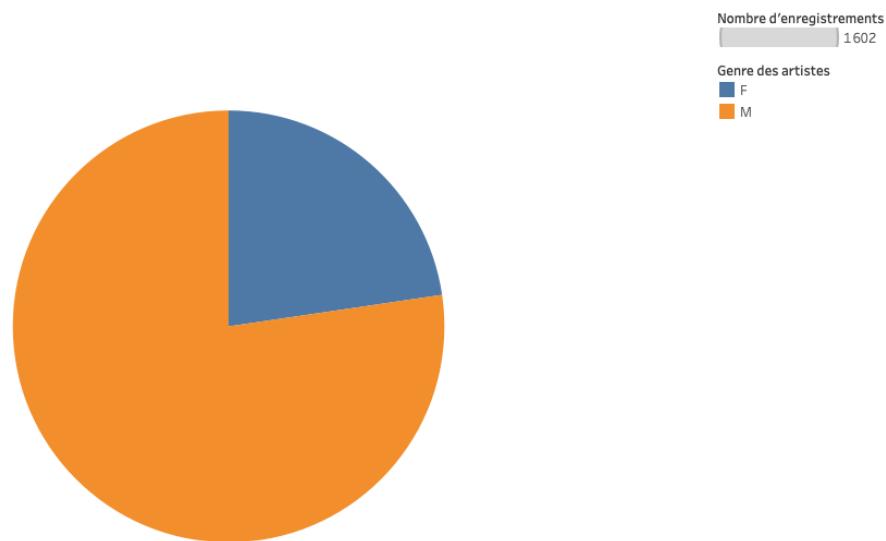


FIGURE 9.3 – Proportion des artistes femmes au salon de la Société des artistes indépendants de 1923.

femmes mécènes⁵.

Même s'il s'appuie sur un petit jeu de données, cet exemple de visualisation permet d'aborder l'histoire de l'art sous un angle quantitatif tandis que l'étude de la participation

5. C. Dossin et H. Alkema, « Women Artists Shows · Salons · Societies..., p. 8.

des artistes femmes à certaines expositions tend à allier l'histoire de l'art à l'étude des genres. Pour élargir le propos et tirer des conclusions plus pertinentes, il serait intéressant d'interroger un nombre plus important de catalogues d'exposition et de croiser les résultats obtenus avec des sources historiques.

9.1.2 Premières perspectives

L'un des principaux objectifs du *workflow* est de pouvoir augmenter de façon majeure la rentrée des données dans BasArt. L'automatisation de la récupération des entrées de catalogues permet un gain de temps considérable puisque la rentrée de la majorité des données n'est plus à faire manuellement⁶. Cela permettra notamment aux membres d'Artl@s d'insérer les données des expositions où des milliers d'œuvres ont été recensées. Cette automatisation peut également inciter les institutions partenaires à collaborer au projet et à envoyer leurs données directement dans BasArt ou sur le GitLab de Béatrice Joyeux-Prunel.

D'autre part, l'insertion de nouvelles données dans BasArt va offrir aux chercheur·se·s et aux autres utlisateur·rice·s de la base de données la possibilité de consulter des informations concernant de nouvelles expositions. L'accès à un plus grand nombre de données va également leur permettre de les interroger et de les comparer afin de les étudier.

L'usage de la base de données restera sensiblement le même et servira autant aux étudiant·e·s, qu'aux chercheur·se·s ou encore aux professionnel·le·s du monde l'art. Malgré cela, le projet Artl@s va prochainement proposer à ses utilisateur·rice·s de nouvelles fonctionnalités. Tout d'abord, un serveur IIIF rassemblera des catalogues d'exposition mais également des périodiques ; il sera possible de les visionner publiquement ou en privé⁷ sur le visualiseur Mirador. Celui-ci présente l'avantage de pouvoir afficher dans une interface commune des documents compatibles avec les protocoles IIIF provenant de divers dépôts. Par ailleurs, l'équipe Artl@s est en train de déployer un volet de recherche dédié à la reconnaissance des images présentes dans les catalogues d'exposition et les périodiques pour en étudier leur circulation. L'intégration de ces deux projets dans Artl@s va permettre d'aller plus loin dans l'étude de la circulation des œuvres puisqu'il sera désormais possible de suivre une œuvre dans les expositions mais également dans les périodiques. Ces nouvelles données vont offrir la possibilité d'étudier la réception d'un artiste tant à travers sa participation à des expositions qu'à son apparition dans des publications artistiques.

6. Certaines informations, comme le genre de l'artiste, demeurent néanmoins à renseigner manuellement lorsqu'elles ne sont pas précisées dans le catalogue.

7. Certains catalogues ne pourront pas être visionnés en public à cause de droits de diffusion des données imposés par certaines institutions.

9.2 Quelles améliorations pour un *workflow* entièrement *open source* ?

9.2.1 Le choix de l'*open source*

Le choix de réaliser un *workflow open source* utilisant des logiciels eux-mêmes *open source* est une volonté émanant des membres du projet Artl@s qui utilisent déjà des technologies libres. L'*open source*, comme le rappelle Amel Charleux dans sa thèse, se fonde sur une philosophie d'ouverture, de partage et de diffusion⁸. Cette idée est soulignée dans la description de la base de données BasArt :

BasArt est une base collaborative de catalogues d'expositions du monde entier, du XIXe siècle à nos jours. Elle est libre d'accès, disponible pour tous, avec une interface cartographique et statistique simple destinée à faciliter les recherches à l'échelle mondiale. Son but est de décentrer progressivement les sources à disposition des chercheur.se.s, donc de décentrer leur regard souvent focalisé sur l'Europe et l'Amérique du Nord⁹.

L'utilisateur·rice se retrouve au cœur des préoccupations du projet Artl@s mais aussi des principes de l'*open source*. En effet, l'*Open Source Initiative*, organisation qui promeut l'*open source*, prône une approche *user centric*, c'est-à-dire focalisée sur les utilisateur·rice·s d'un site. La volonté d'Artl@s de diffuser des données, jusqu'alors dispersées¹⁰, au plus grand nombre, mais aussi de partager les étapes du *workflow* rejoint l'idée de proposer un contenu réutilisable par les chercheur·se·s en histoire de l'art mais aussi en humanités numériques. Le projet invite également les contributeur·rice·s à y collaborer, un geste qui, selon Amel Charleux, complexifie les usages de l'*open source*¹¹. En outre, les utilisateur·rice·s peuvent consulter des données, les télécharger sous forme de CSV, accéder au *workflow* et y contribuer. La charte de la licence Creative Commons CC-BY, choisie pour le *workflow*, autorise d'ailleurs le partage et la modification des données à condition que le travail soit cité, s'il est réemployé par autrui. L'utilisateur·rice est donc libre d'utiliser les données et les technologies déployées pour des usages personnels, un des principes fondamentaux de l'*open source*¹².

D'autre part, les valeurs de l'*open source* rejoignent celles de la valorisation patrimoniale¹³ qui cherche avant tout à diffuser auprès du plus grand nombre des données, sous

8. Amel Charleux, *L'OPEN SOURCE ENTRE CONCURRENTS - Approche de la création et de l'appropriation de valeurs par les business models et la coopération*, dir. Anne Mione, Montpellier, Université de Montpellier, 2019, p. 12.

9. « BasArt : une base mondiale de catalogues d'expositions...

10. Les données diffusées dans BasArt sont issues de catalogues d'exposition, disponibles ou non en ligne, provenant de diverses institutions patrimoniales.

11. A. Charleux, *L'OPEN SOURCE ENTRE CONCURRENTS...*, p. 13.

12. *ibid.*, p. 17.

13. *ibid.*, p. 14.

diverses formes. BasArt a pour objectif de rassembler des données propres aux expositions des XIX^e et XX^e siècles dans le but de les valoriser par la recherche et l'étude. En offrant l'accès à des catalogues d'exposition parfois inaccessibles, de par leur dispersion sur internet ou leur absence¹⁴, Artl@s offre aux chercheur·se·s mais aussi aux amateur·rice·s d'art la possibilité de consulter des informations sur les expositions, les exposant·e·s et les œuvres exposées, le tout à travers une interface cartographique et chronologique.

L'*open source* garantit tout autant la transparence des données, leur interopérabilité ainsi que l'indépendance technologique¹⁵. Cette dernière idée est importante pour le projet Artl@s qui souhaite dépendre le moins possible de logiciels privés, dont l'accès est généralement soumis à des frais d'abonnement. En effet, même si Artl@s et Visual Contagions reçoivent actuellement des financements de la part de diverses institutions, l'équipe a préféré utiliser des logiciels libres, ne nécessitant pas d'abonnement ou de licences, afin de privilégier la pérennité du projet¹⁶.

L'*open source* présente donc plusieurs intérêts pour un projet universitaire tel Artl@s. Il permet à la fois de partager et de diffuser des données librement, de favoriser la contribution des utilisateur·rice·s en leur donnant accès aux outils utilisés et de garantir l'indépendance et la pérennité du projet.

9.2.2 Quelles améliorations ?

Si le projet Artl@s et le *workflow* ont été pensés de façon à être entièrement *open source*, certaines étapes de ce dernier dépendent de logiciels impliquant des coûts. C'est le cas de l'éditeur Oxygen dont la licence doit être renouvelée tous les ans, tant pour les entreprises, les universités que les particulier·e·s¹⁷, même si ces dernier·e·s peuvent télécharger une version d'essai valable 30 jours. Dans la mesure où le projet Artl@s bénéficie désormais du soutien de la chaire d'Humanités numériques de l'UNIGE, il serait possible d'obtenir une licence universitaire d'Oxygen pour les besoins du *workflow*. Malgré son coût élevé, ce logiciel a fait ses preuves dans les milieux universitaires et est également utilisé par les membres de l'équipe e-editiones¹⁸ dont le *workflow* permet d'encoder des catalogues de vente de manuscrits. Il est toutefois possible de remplacer Oxygen par un autre éditeur XML, mais l'intérêt qu'offre le projet .xpr sera perdu puisque tous les logiciels ne proposent pas cette option.

Par ailleurs, l'équipe de la coopérative READ, fondatrice de Transkribus, a annoncé

14. Certaines données proviennent de catalogues d'exposition qui n'ont pas été numérisés.

15. Nordine Benkeltoum, « Open source et systèmes critiques : le cas Thales », *17ème Colloque de l'AIM*, Bordeaux, 2012, p. 5.

16. B. Joyeux-Prunel, « Bases de données et gestion de projets en humanités numériques...

17. « Buy now », *Oxygen*, <https://www.oxygenxml.com/buy.html> (visité le 23/09/2020).

18. Lucie Rondeau du Noyer et Simon Gabay, « DOCUMENTATION », *katabase/GROBID_Dictionaries*, https://github.com/katabase/GROBID_Dictionaries/blob/master/DOCUMENTATION.md (visité le 23/09/2020).

au mois de juin 2020 que les services proposés par l’application allaient devenir payants¹⁹. Cette décision a récemment été confirmée dans la *newsletter* de Transkribus : la tarification du traitement des pages sera effective à partir du 19 octobre 2020. Les tarifs vont dépendre de plusieurs facteurs, notamment du type de document à traiter, l’OCRisation des manuscrits sera plus coûteuse que celle des imprimés. Même si les catalogues d’exposition sont des sources imprimées, le nombre important de pages à transcrire²⁰ deviendrait coûteux pour Artl@s puisque les services gratuits vont rester limités à un petit nombre de pages²¹. Une des solutions envisagée par l’équipe-projet ALMAaCH ainsi que par les membres d’e-ditiones et d’Arts@ est d’utiliser l’application d’OCR Kraken, qui a l’avantage d’être *open source*. Elle a été développée par Benjamin Kiessling entre les universités de Leipzig et PSL²² et, contrairement à d’autres logiciels d’OCR, Kraken s’appuie sur un réseau de neurones artificiels qui apprend à reconnaître les lettres avant d’apprendre à segmenter la page en zones, puis en lignes et enfin en mots. Ce dernier point suppose d’entraîner un segmenteur de lignes²³, ce qui nécessite d’annoter de nombreuses données comme l’a soulevé Laurent Romary lors d’une réunion entre les personnes travaillant avec *GROBID-dictionaries*²⁴. Par ailleurs, les membres de l’équipe-projet ALMAaCH se sont également aperçus que les fichiers ALTO produits par Kraken sont différents de ceux produits par Transkribus. L’enjeu est donc pour elles et eux de trouver les différences entre ces fichiers ALTO et faire en sorte que ceux en sortie de Kraken soient compatibles avec ceux lus par *GROBID-dictionaries*. La résolution de ce problème permettrait à toutes les équipes travaillant sur le logiciel d’encodage automatique des catalogues et des dictionnaires de travailler à partir d’un *workflow* totalement indépendant de Transkribus. Outre le besoin d’entraîner un segmenteur, le principal inconvénient de Kraken est qu’il s’utilise via le terminal et est donc moins accessible que Transkribus qui possède une application avec interface graphique. Si les membres de l’Inria arrivent à obtenir des fichiers ALTO compatibles avec *GROBID-dictionaries* et à remplacer Transkribus par Kraken dans le *workflow*, il serait utile qu’ils ou elles écrivent un tutoriel à l’intention des utilisateur·rice·s, tel·le·s les membres d’Arts@, pour les guider dans le maniement des lignes de commande.

19. « What will change in 2020 ? », *READ COOP*, <https://readcoop.eu/transkribus-pricing/> (visité le 23/09/2020).

20. Les catalogues d’exposition peuvent faire plus de 200 ou 300 pages et des centaines de catalogues sont encore à intégrer dans BasArt.

21. Philippe Chassignet et Emmanuelle Perrin, « Rendre visible la face cachée de l’iceberg », *ArchéOrient - Le Blog*, <https://archeorient.hypotheses.org/14807> (visité le 30/07/2020).

22. mittagessen, *kraken*, GitHub, <https://github.com/mittagessen/kraken> (visité le 23/09/2020).

23. Benjamin Kiessling, *Kraken - an Universal Text Recognizer for the Humanities*, <https://dev.clariah.nl/files/dh2019/boa/0673.html> (visité le 04/08/2020).

24. Cette réunion s’est déroulée le 1er juillet 2020, y étaient présents plusieurs membres de l’équipe-projet ALMAaCH – dont Laurent Romary, Alix Chagué et Mohamed Khemakhem – Simon Gabay et moi-même.

Conclusion

Le présent mémoire s'est attaché à retracer les différentes étapes d'un *workflow* permettant d'encoder automatiquement en XML-TEI les entrées des catalogues d'exposition, dans le but d'alimenter la base de données BasArt. Il a également été l'occasion de revenir sur les enjeux des fichiers ALTO, récupérés en sortie de l'OCR, au sein de la chaîne de traitement. Les différentes missions qui m'ont été confiées m'ont permis de réfléchir à la modélisation de l'encodage des catalogues d'exposition mais aussi des sources catalographiques en général, de tester deux versions d'un logiciel de *machine learning* – en l'occurrence *GROBID-dictionaries* – et de proposer une chaîne de traitement, inspirée du précédent travail de Lucie Rondeau du Noyer, accompagnée de tutoriels à destination des membres du projet Artl@s.

Avant même de s'intéresser à la chaîne de traitement réalisée par Lucie Rondeau du Noyer pour le projet e-ditiones, il importait de proposer un modèle d'encodage XML-TEI répondant aux besoins du projet Artl@s. Cet encodage devait à la fois rendre compte de la structure des entrées des catalogues d'exposition mais aussi offrir la possibilité d'extraire certaines informations assez finement pour qu'elles puissent concorder avec les entrées de BasArt. Nous nous sommes arrêté·e·s sur un modèle malléable, qui permet d'encoder les entrées jusqu'à un certain niveau, correspondant à ceux de *GROBID-dictionaries*, et d'affiner l'encodage si besoin. L'ajout de balises récupérant des informations précises, comme la date de l'œuvre, sa technique ou ses dimensions, offre la possibilité aux membres du projet d'ajouter une étape supplémentaire au *workflow* ce qui permettrait d'extraire automatiquement ces informations. Ce travail sur l'encodage XML-TEI des catalogues d'exposition a abouti à la proposition d'un encodage XML-TEI pour les catalogues en général dans le but de standardiser la structuration de ces sources. Ces réflexions ont fait suite à celles entamées par Laurent Romary qui souhaite développer un *fork* de *GROBID-dictionaries* dédié aux sources catalographiques : *GROBID-cat*. Ce dernier travail de modélisation m'a également permis de saisir les enjeux d'un projet de recherche inter-institutionnel. Dans le cas de la recherche sur l'encodage automatique des catalogues grâce à *GROBID-dictionaries*, il importe de tester le logiciel sur différentes sources, telles les catalogues de vente de manuscrits et les catalogues d'exposition, afin d'observer le comportement du logiciel.

Suite à la volonté de Béatrice Joyeux-Prunel, nous avons préparé une campagne

d’entraînement pour observer le comportement de *GROBID-dictionaries* avec les catalogues d’exposition. L’application pouvant lire un nouveau format de fichiers, nous nous sommes appliquées, avec Ljudmila Petkovic, à récupérer la transcription des catalogues en format ALTO. Celui-ci nous a permis d’insérer des informations supplémentaires dans le texte OCRisé²⁵, comme l’information typographique et la taille des caractères. Pour parvenir à récupérer ces informations, Ljudmila Petkovic et Simon Gabay ont entraîné un modèle HTR sur le logiciel Transkribus qui balise l’information typographique dans la transcription. Tous les deux ont également écrit un script qui permet d’évaluer la performance de ce modèle. La comparaison des données de Ljudmila Petkovic, produites à partir de catalogues de vente de manuscrits, et de celles que j’ai produites à partir de catalogues d’exposition a montré que le modèle HTR était plus efficace sur des sources bien numérisées dont la mise en page est plus aérée.

Le retard de la livraison de la dernière version de *GROBID-dictionaries* a entraîné le remaniement de mes missions et un retour vers le format PDF, seul type de document lisible par l’application. La recherche de solutions pour convertir les fichiers ALTO, tout en conservant l’information typographique dans le PDF nous a amené·e·s à choisir d’opérer cette transformation en passant par XSL-FO et non par un script python. Ce choix a été fait en fonction de deux facteurs : la facilité de lancement du script XSL-FO dans l’éditeur XML Oxygen et le temps supplémentaire que m’aurait pris l’écriture de plusieurs scripts python. Cette étape, ajoutée dans un second temps au *workflow*, s’est avérée être bénéfique puisqu’elle a permis de démontrer que les modèles entraînés à partir de PDF contenant l’information typographique sont plus performants que les modèles entraînés à partir de fichiers ALTO. En testant la nouvelle version de *GROBID-dictionaries*, j’ai pu participer activement à un projet scientifique en cours et émettre de premières conclusions sur son fonctionnement. Les résultats des différents modèles entraînés à partir de fichiers PDF et ALTO ont démontré que l’ajout de l’information typographique favorise l’amélioration des modèles : le logiciel de *machine learning* a bien intégré les nouvelles *features* indiquant si le texte est dans une police différente – c’est-à-dire en gras ou en italique – ou pas. Par ailleurs, la comparaison des résultats a dévoilé que ceux-ci étaient plus performants lorsque les modèles étaient entraînés à partir de fichiers PDF plutôt qu’à partir de fichiers ALTO. Les raisons de ces résultats restent encore inconnues même s’il est possible d’émettre des hypothèses : il est probable que la multitude d’informations contenues dans les fichiers ALTO les rendent moins lisibles pour *GROBID-dictionaries*, d’autant plus qu’il a été codé pour traiter des fichiers PDF.

Enfin, la réalisation du *workflow* a soulevé plusieurs problématiques. Il devait à la fois être réutilisable par les membres du projet Artl@s ainsi que ses partenaires, et répondre aux critères de l’*open source*. La principale difficulté a été de rendre la chaîne de

25. Jusqu’alors, le texte OCRisé était récupéré dans un fichier PDF dans lequel seule la position des mots sur la page était indiquée.

traitement la plus « *user friendly* » possible puisque les membres d'Artl@s sont principalement formé·e·s aux bases de données et non aux lignes de commande et au langage XML-TEI. Pour pouvoir les accompagner au mieux, toutes les étapes du *workflow* ont été détaillées et illustrées de captures d'écran. Malgré cela, je reste consciente qu'il est nécessaire de prévoir une ou plusieurs formations pour que les membres du projet s'approprient le *workflow*, ne serait-ce que pour apprendre à maîtriser les lignes de commande afin de lancer *GROBID-dictionaries*. D'autre part, le souhait de l'équipe de travailler à partir d'outils *open source* a été compromis par l'annonce de la coopérative READ, créatrice de Transkribus, qui est en train de remplacer ses services gratuits par des services payants. La durée du stage ne m'a pas permis de proposer une solution viable, néanmoins, les autres équipes utilisant *GROBID-dictionaries* sont en train de se tourner vers le logiciel d'OCR Kraken en vue d'être indépendantes de Transkribus. Ce changement de logiciel soulève quelques enjeux, comme l'entraînement d'un segmenteur et de nouveaux modèles.

Désormais, il importe d'avoir un retour utilisateur·rice sur le *workflow* afin d'évaluer son accessibilité et les points qui nécessitent d'être approfondis, soit en éclaircissant certaines étapes soit en proposant des formations adéquates et adaptées aux besoins des membres du projet. Par ailleurs, les dernières semaines du stage ont été consacrées à la production de données à travers l'encodage des catalogues de la Société des artistes indépendants. Cette dernière tâche permet de valider la fonctionnalité du *workflow* et de générer des fichiers XML-TEI qui sont ensuite transformés en CSV, dans le but d'être insérés dans BasArt. Pour le moment, ces fichiers nécessitent une relecture et une redistribution des données dans les colonnes correspondantes du CSV, puisque les informations n'ont été encodées que jusqu'à un certain niveau. D'ailleurs, certaines d'entre elles ne sont pas toujours renseignées dans les catalogues, comme le genre de l'exposant·e et sont donc à rajouter à la main. Ce dernier travail de relecture sera effectué par d'autres membres du projet, suivant le mode de fonctionnement d'Artl@s.

En définitive, les trois premiers mois du stage m'ont permis d'élaborer un *workflow* fonctionnel accompagné de tutoriels à destination de ses futur·e·s utilisateur·rice·s. Certaines étapes peuvent être améliorées, comme celle de l'OCRisation puisque pour le moment, le modèle HTR est principalement efficace sur les sources en langue française. De plus, il est également possible d'entraîner de nouveaux modèles dans *GROBID-dictionaries* adaptés à des types de catalogues autres que ceux des biennales et de la Société des artistes indépendants. Enfin, une étape supplémentaire pourrait être ajoutée pour encoder en XML-TEI certaines informations précises, à l'aide d'un logiciel NER, d'expressions régulières ou encore d'un script python. Ceci permettrait de découper la biographie de l'artiste et la description de l'œuvre afin d'automatiser la récupération de ces informations dans le fichier CSV et en alléger ainsi sa relecture.

Annexes

Annexe A

Sources : entrées de catalogues

A.1 Catalogues d'exposition monographique

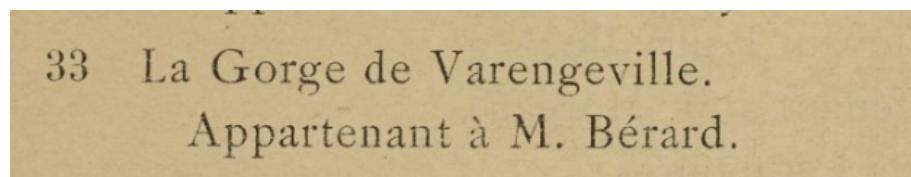


FIGURE A.1 – Exemple d'entrée du *Catalogue de l'exposition des œuvres de Claude Monet. 9 Boulevard de la Madeleine. Ouverte du 1er au 25 mars [1883]*, Paris, 1883.

119. — Le Lac.

Signé à gauche : G. Courbet.
Sans date.

T. — H. 0.70. L. 0.81.

Appartient à M. Barbedienne.

FIGURE A.2 – Exemple d'entrée du catalogue *Exposition des œuvres de Gustave Courbet à l'école des Beaux-Arts (mai 1882)*, avec la coll. de Jules-Antoine Castagnary, Paris, 1882.

A.2 Catalogues de biennales

MARILIA GIANNETTI TORRES (1925)

- 205 COMPOSIÇÃO 4, 1953. 73 x 54.**
206 COMPOSIÇÃO 5, 1953. 61 x 64.

FIGURE A.3 – Exemple d'entrée du catalogue *II bienal de São Paulo, Museu de arte moderna*, dir. Museu de arte moderna et Biennale internationale de São Paulo, São Paulo, Brésil, 1953.

Cannicci Niccolò

- 1 Maternitas (trittico).** (Cornice dell'arch. A. Salvetti).
- 2 Etruria.** .
- 3 All'ovile — notte (pastello).**

FIGURE A.4 – Exemple d'entrée du catalogue *VI. Esposizione Internazionale d'Arte Della Città Di Venezia, 1905 : Catalogo Illustrato*, Venezia, 1905.

A.3 Catalogues de Salons et de musées

Entrées sans gras ni italienique :

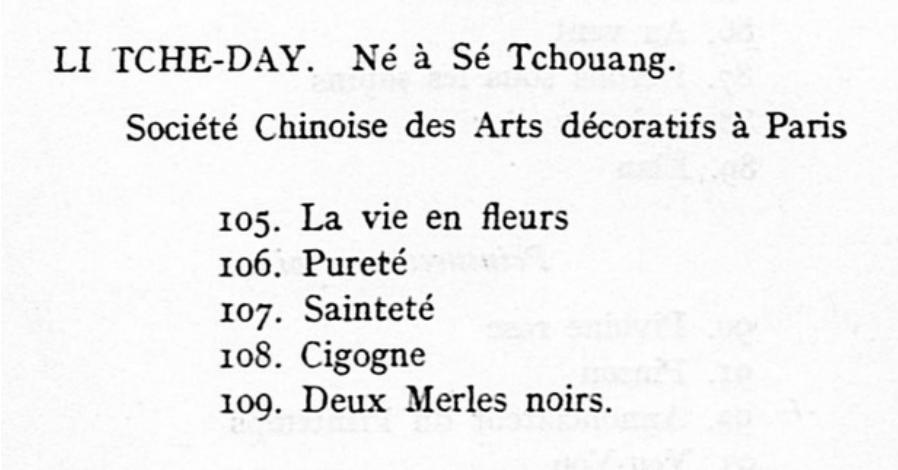


FIGURE A.5 – Exemple d'entrée du catalogue de l'*Exposition chinoise d'art ancien et moderne / organisée sous le patronage du commissariat général de la République à Strasbourg et du Ministre plénipotentiaire de Chine en France, Palais du Rhin, mai-juillet 1924*, avec la coll. de Tsai Yen-Pei, Strasbourg, 1924.

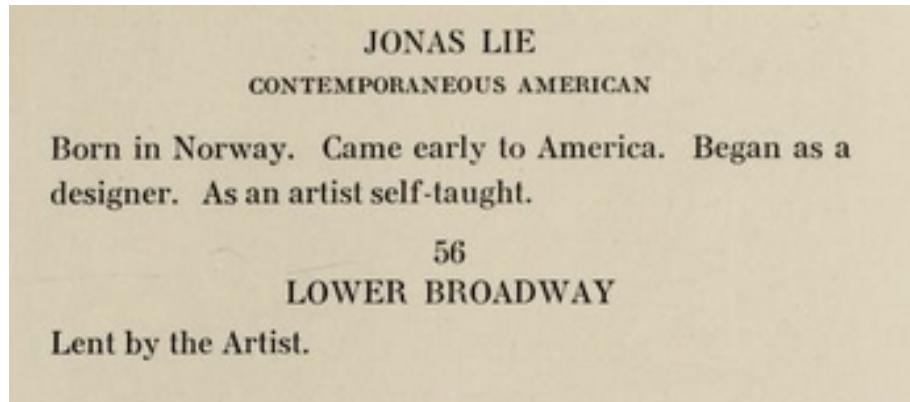


FIGURE A.6 – Exemple d'entrée du *Catalogue of the inaugural exhibition, January seventeenth to February twelfth An. Dni. MCMXII*, avec la coll. d'Harold B. Lee, New York, 1912.

Entrées avec biographie en italique :

| | | |
|--------------------------|-------|-----|
| STAHL, Émile. | . | . |
| <i>Schiltigheim.</i> | | |
| 262. Chasseur à l'affût. | | 500 |
| 263. Le rôti | | 500 |

FIGURE A.7 – Exemple d'entrée du *Catalogue de l'exposition des œuvres d'art d'artistes vivants : Strasbourg, Hôtel-de-ville, du 11 mai au 8 juin 1884*, Strasbourg, 1884.

| |
|--|
| M ^{me} CHARMEIL, de Metz, rue d'Assas, à Paris. |
| 9. Bouquet de Marguerites. |
| 10. <i>Idem</i> de Dahlias. |
| 11. Une Jacinthe. |

FIGURE A.8 – Exemple d'entrée du *Catalogue des peintures, miniatures, aquarelles, dessins, sculptures et lithographies exposés à Nancy... par les artistes lorrains*, Nancy, 1847.

Entrées avec numéros en gras :

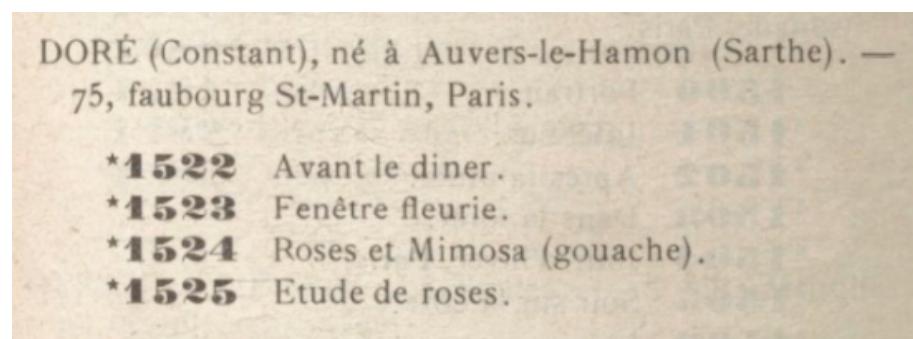


FIGURE A.9 – Exemple d'entrée du *Catalogue de la 22e exposition 1906 : grandes serres de la ville de Paris... du 20 mars au 30 avril... / Société des artistes indépendants, dir. L'Emancipatrice, Paris, 1906.*

Entrées par titre d'œuvre :

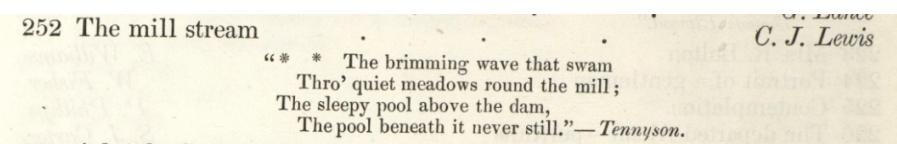


FIGURE A.10 – Exemple d'entrée du catalogue *The exhibition of the Royal Academy of Arts. MDCCCLIX. (1859). The ninety-first*, London, 1859.

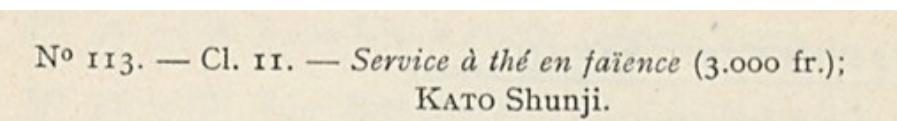


FIGURE A.11 – Exemple d'entrée du *Catalogue illustré de la section japonaise à l'exposition internationale des arts décoratifs et industriels modernes, Paris. 1925.*

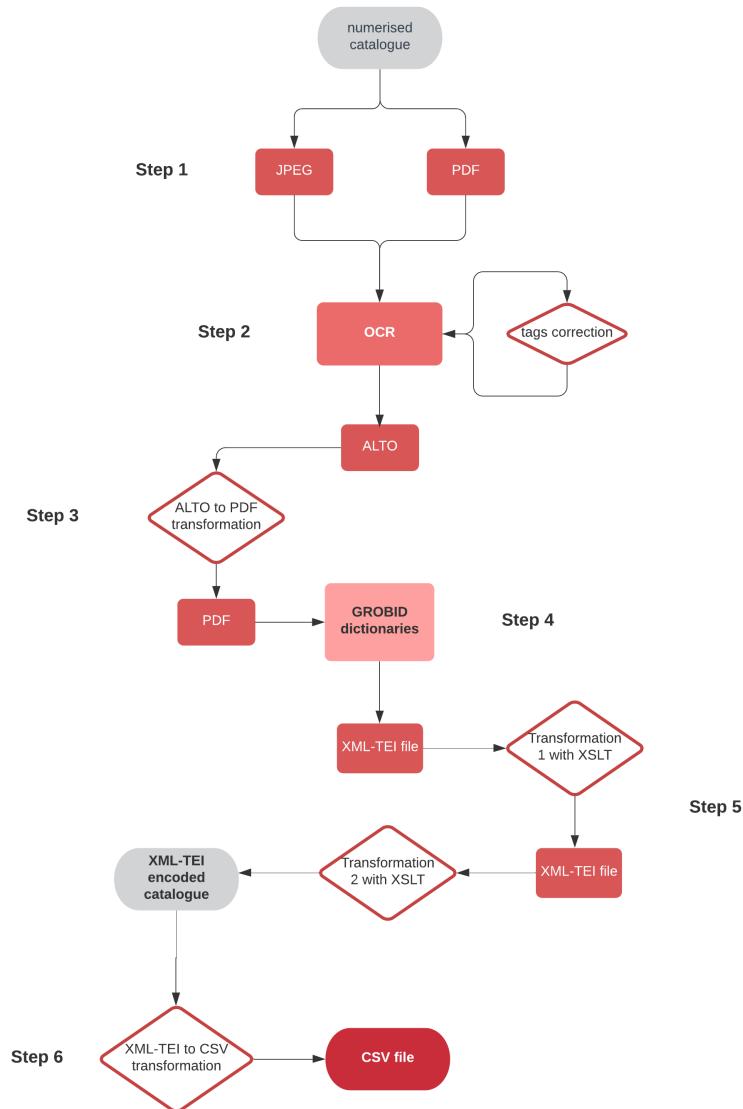
Annexe B

Workflow

The aim of the Artl@s project is to study the global circulation of images from the 1890s to the advent of the Internet. It manages several research projects on artistic and cultural globalization. It also takes an active interest in digital methodologies and in the circulation of images.

One of its projects is BasArt, an online database of exhibition catalogs from the 19th and 20th centuries. To expand the database, Béatrice Joyeux-Prunel decided to find a way to process and structure the scanned exhibition catalogs in order to easily complete the database. They decided to use a pivot XML-TEI format and GROBID-dictionaries, a machine-learning software designed to automatically structure the lexicographical resources and to automatically encode the exhibition catalogs.

This repository includes all the steps to automatically encode an exhibition catalogue for the Artl@s project.



Prerequisites

1. Download this repository. Run in your terminal this command :

```
git clone https://github.com/carolinecorbieres/ArtlasCatalogues.git
```

2. For the steps 1 and 2 you will need to have python 3 installed. Run in your terminal this command : `python -m pip install SomePackage`

- If you want to work in a virtual environment, you have to create one :

- Install the virtualenv PyPI library.

```
pip3 install virtualenv
```

- Move to the project directory.
- `cd YOUR_PATH_TO_THE_PROJECT/visual-contagions/Catalogues`
- Set up your Python virtual environment.
- `virtualenv -p python3 env`
- Activate the environment.
- `source env/bin/activate`
- Install libraries and dependencies : `pip3 install -r requirements.txt`

Contribute

You can contribute to the project by encoding new catalogues and/or creating new models for GROBID-dictionaries. If you do so, please update this repository with your new data.

Thanks

Thanks to Simon Gabay, Matthias Gille Levenson, Béatrice Joyeux-Prunel, Ljudmila Petkovic, Auriane Quoix, Jean-Paul Rehr and Lucie Rondeau du Noyer for their help and work.

Credits

This repository is developed by Caroline Corbières with the help of Simon Gabay, under the supervision of Béatrice Joyeux-Prunel, as part of the project Artl@s¹.

Licence

This repository is CC-BY.

B.1 Step 0 : Converting CSV files into XML-TEI headers

This README is written by Auriane Quoix. You can find all her work on the Visual Contagions project here²

B.1.1 Requirements

The conversion of CSV files into XML-TEI headers is carried out using Oxygen XML Editor. You can download it here³.

1. <<https://artlas.huma-num.fr/fr/>>
 2. <https://github.com/AurianeQuoix/VISUAL_CONTAGIONS>
 3. <https://www.oxygenxml.com/xml_editor/download_oxygenxml_editor.html>

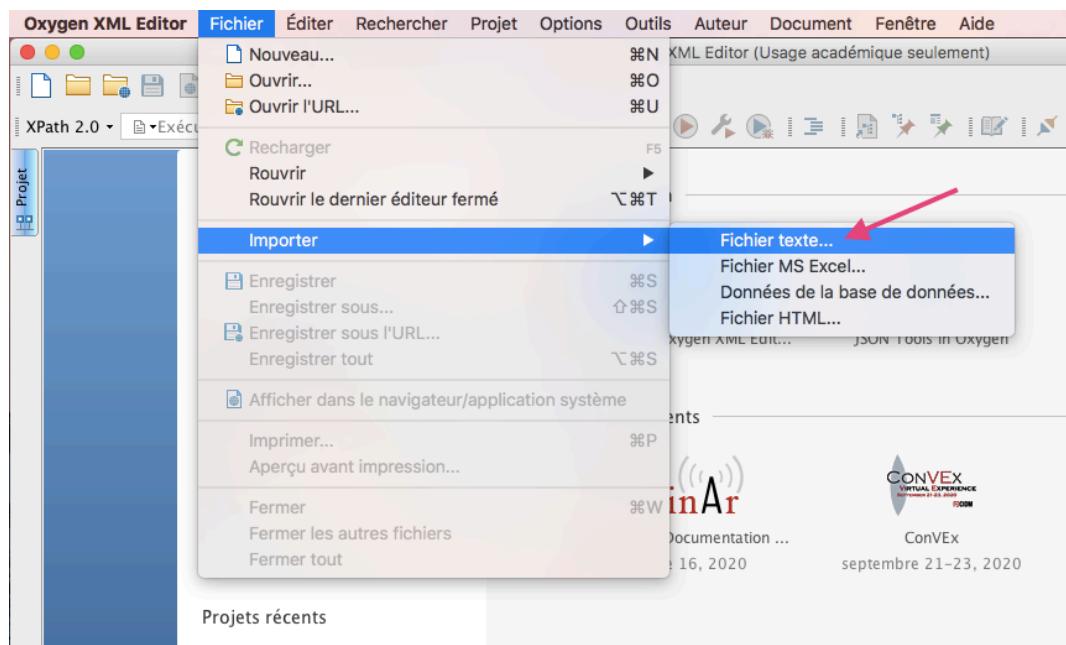
B.1.2 Import of the CSV files into Oxygen XML Editor

It is possible with Oxygen XML Editor to import text files as XML documents (it concerns txt or csv extensions). So we will do this with the two spreadsheets published here and here (click on the links to download the two CSV files).

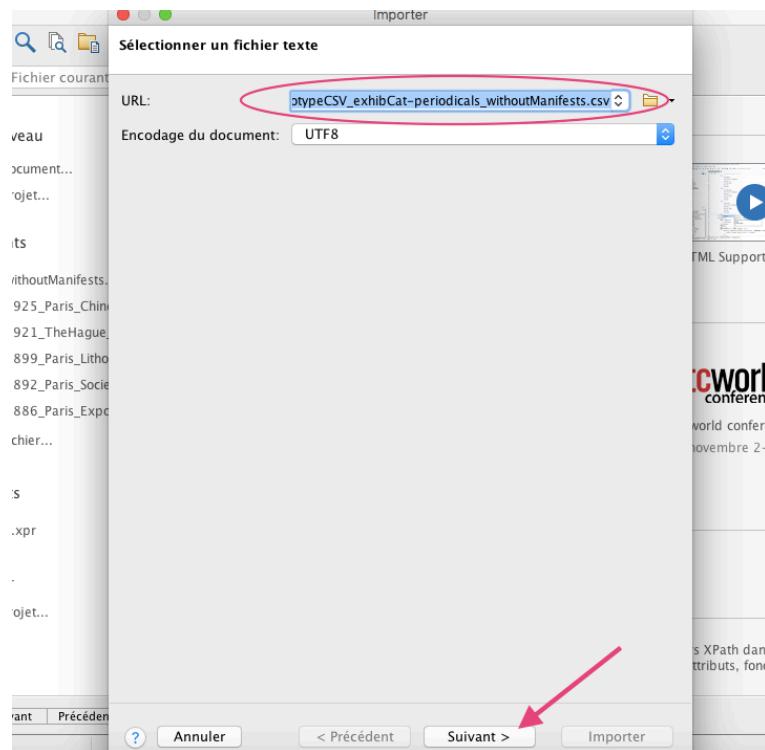
The first one concerns the resources without IIIF manifests and the second one concerns those for which it already exists a manifest.

After downloading the CSV files, you must open Oxygen XML Editor.

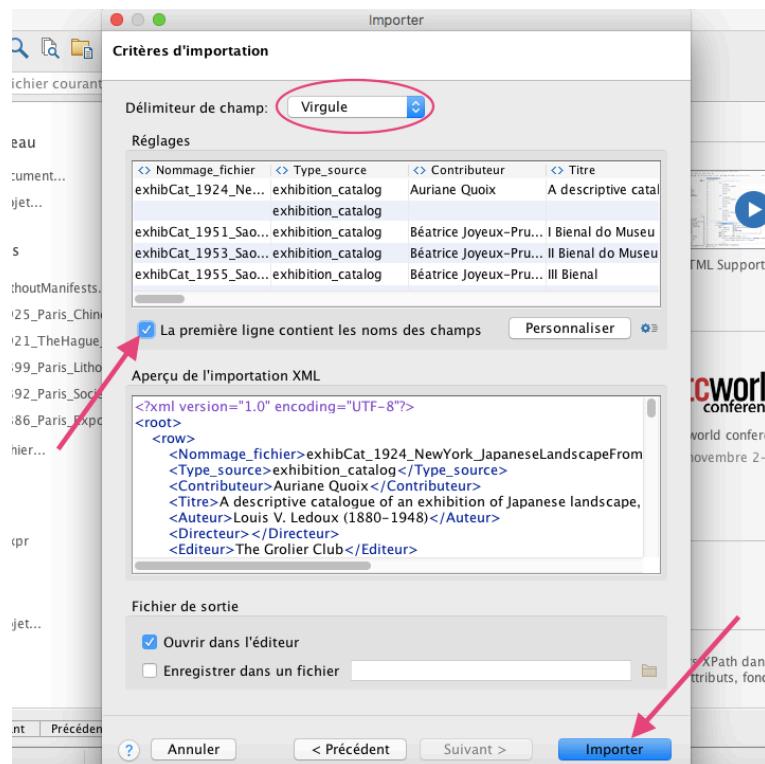
1. Go to File > Import > Text File.



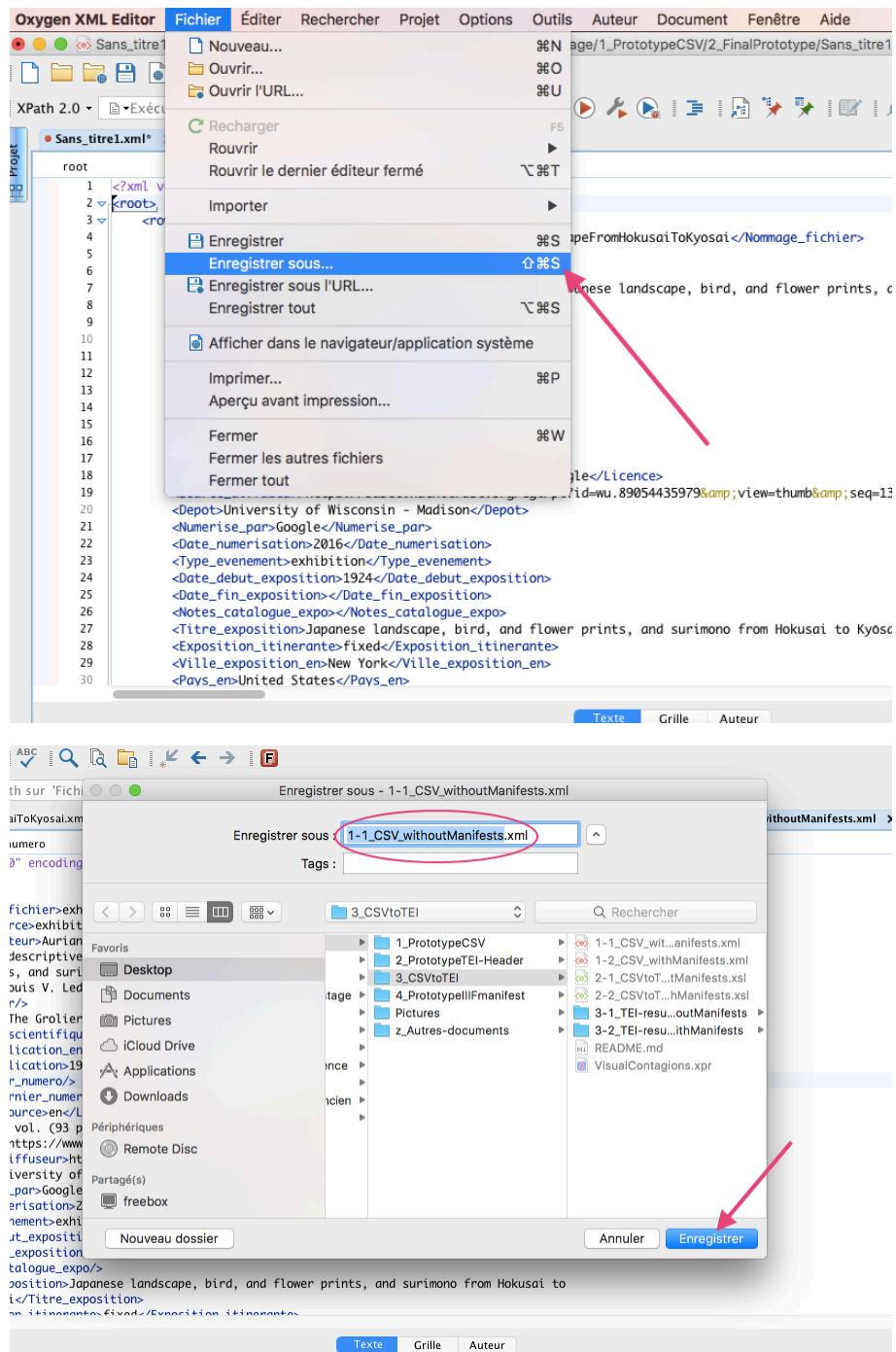
2. In the dialog box, select the URL of the first CSV file downloaded (PrototypeCSV_exhibCat-periodicals - withoutManifests), click on Open and then on the Next button.



3. Now, the import criteria dialog box is displayed. - Verify that the selected field delimiter for the import settings is **Comma**. - Select the option **First row contains field names**. - Click on the **Import** button.



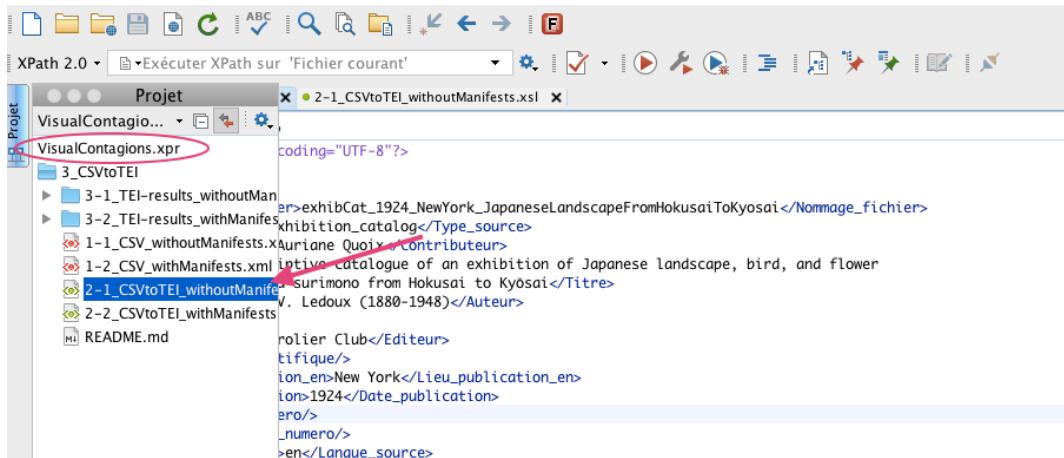
4. The new XML document is opened in the editor. Go to **File > Save** and give this name to the file : "1-1__ CSV__withoutManifests.xml".



You must follow the same procedure to import the second CSV file (PrototypeCSV_exhibCat-periodicals - withManifests) into Oxygen XML Editor. Once you get to 4., name it : "1-2__ CSV_withManifests.xml".

B.1.3 Generation of TEI headers with XSLT

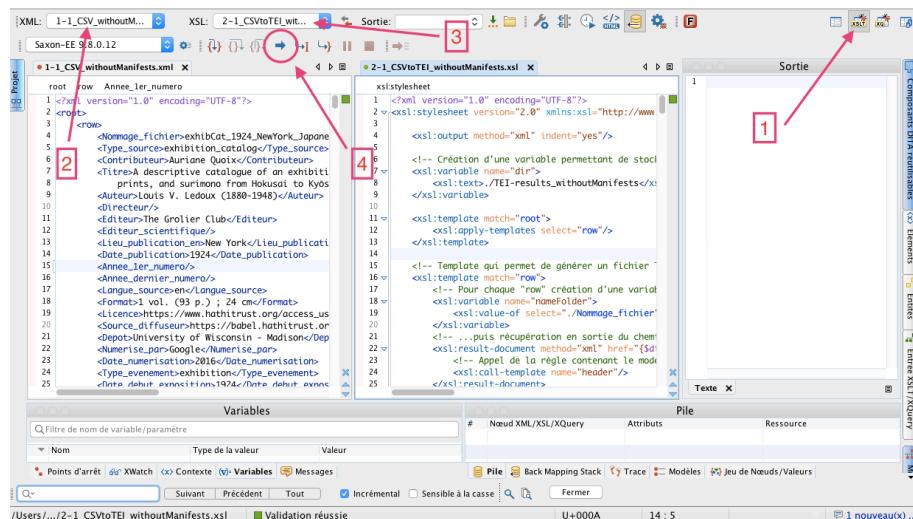
1. Open the ****2-1__ CSVtoTEIwithoutManifests.xsl** file (VISUALCONTAGIONS/3_CSVtoTEI/2-1 __CSVtoTEI _withoutManifests.xsl)** in Oxygen. Or open the **Visual-Contagions.xpr** project and click on **2-1__ CSVtoTEI__withoutManifests.xsl** to open it.



2. Now the transformation can be launched. - Open the XSLT Debugger perspective. For that, select the XSLT Debugger button in the top-right corner of the interface (or go to Window > Open perspective > XSLT Debugger). On the picture, it corresponds to the arrow number one.

- Select the XML file **1-1_ CSV_withoutManifests.xml** (arrow number 2) and the XSL file **2-1_ CSVtoTEI_withoutManifests.xsl** (arrow number 3).
- Click on the run button (arrow number 4).

The results are in the **3-1_ TEI-results_withoutManifests** folder.



3. You can follow the same procedure to transform the other file.

- After opening them, select the XML file **1-2_ CSV_withManifests.xml** and the XSL file **2-2_ CSVtoTEI_withManifests.xsl** in the XSLT Debugger perspective.
- Click on the run button.

The results are in the **3-2_ TEI-results_withManifests** folder.

4. Move the files in the corresponding folder.

- In the Catalogues folder, create a new folder named `exhibCat_NAME_OF_THE_CATALOGUE` and another one named TEI included in the first one.
- Then move your `exhibCat_NAME_OF_THE_CATALOGUE_header.xml` file in the TEI folder.

B.2 Step 1 : Import catalogue pages or PDF

To process catalogues into the workflow, we need to import catalogue pages. For the use of the OCR software, it is better to have high resolution images but it also works with PDF documents.

B.2.1 Digitised Catalogues

Catalogue Images using IIIF

If the digitised catalogue have images using IIIF, you can get image pages thanks to the script `import_iiif.py`. Its arguments are based on the URI Manifest model : `{scheme}://{host}/{prefix}/{identifier}/manifest`. The default arguments are set on the Gallica IIIF API, however you can change that to adapt them to any IIIF API.

1. If you want to download a catalogue from Gallica, run in the terminal the following commands :

- Go to your folder.

```
cd YOUR_PATH_TO_THE_FOLDER/visual-contagions/Catalogues/1_ImportCatalogues
```

- Copy the ark identifier of the catalogue in Gallica.

The screenshot shows the BnF Gallica interface. On the left, there's a sidebar with various icons. The main area has a title bar with 'BnF Gallica' and a search bar. Below that, there are navigation links: 'TOUT GALlica', 'Rechercher...', 'TOUTES NOS SÉLECTIONS', 'PAR TYPES DE DOCUMENTS', 'PAR THÉMATIQUES', and 'PAR AIRES GÉOGRAPHIQUES'. The main content area displays a catalog entry for 'Société des artistes indépendants. 6, Catalogue des œuvres exposées 1890 ... Salon des indépendants...'. On the left, a sidebar titled 'Informations détaillées' shows a 'NOTICE' section with detailed information about the catalog, including its title, author, date, type, and rights. The 'Identifiant' field is highlighted with a red arrow and contains the Ark URL 'ark:/12148/bpt1k933463t'. To the right of the sidebar is a large thumbnail image of a page from the catalog, which is a black and white photograph of a document with the year '1890' printed on it and a circular stamp that reads 'FONDATION SMITH-LESQUEP'.

- Run the script with the ark identifier argument. If you need some help, you can run the script with -h argument. `python3 import_iiif.py ark:/ARK_IDENTIFIER`
- The script will download pages in JPEG in a `download/IDENTIFIER_NAME` folder.

2. If you want to download a catalogue from another IIIF API, run in the terminal the following commands :

- Go to your folder.

```
cd YOUR_PATH_TO_THE_FOLDER/visual-contagions/Catalogues/1_ImportCatalogues
```

- Find the IIIF manifest of the catalogue, for example :

```
https://bibliotheque-numerique.inha.fr/iiif/21124/manifest.
```

- Run the script by substitute Gallica arguments for the corresponding arguments of the URI. If you need some help, you can run the script with -h argument. `python3 import_iiif.py -s SCHEME -d DOMAIN -p PREFIX IDENTIFIER -m MANIFEST`
- You can call any argument with the following prefix :

- `-s` to substitute the scheme of the URI, by default it is `https`
- `-d` to substitute the domain (hostname) of the URI, by default it is `gallica.bnf.fr`
- `-p` to substitute the prefix of the URI, by default it is `iiif`
- `-m` to substitute the manifest name of the URI, by default it is `manifest.json`
- You always must indicate the identifier argument.

For example, for the following URI (`https://bibliotheque-numerique.inha.fr/iiif/21124/manifest`), you should run : `python3 import_iiif.py -d bibliotheque-numerique.inha.fr 21124 -m manifest`

- The script will download pages in JPEG in a `download/IDENTIFIER_NAME` folder.

3. Create a folder for the catalogue named `exhibCat_NAME_OF_THE_CATALOGUE` in `Catalogues` folder. The name of the catalogue takes 3 arguments : the date, the location and the name of the exhibition. Here are some examples : `exhibCat_1892_Paris_SocieteArtistesIndependants`, `exhibCat_1921_TheHague_ExpositionHollandaise` and `exhibCat_1965_Paris_Biennale`.

4. When you have all your pages in JPEG, you can move them in a `JPG` folder that you create in the corresponding `exhibCat_NAME_OF_THE_CATALOGUE` folder.

PDF Catalogue

If the digitised catalogue only exists in PDF :

1. Import the PDF.

2. Create a folder for the catalogue named `exhibCat_NAME_OF_THE_CATALOGUE` in `Catalogues` folder. The name of the catalogue takes 3 arguments : the date, the location and the name of the exhibition. Here are some examples : `exhibCat_1892_Paris_SocieteArtistesIndependants`, `exhibCat_1921_TheHague_ExpositionHollandaise` and `exhibCat_1965_Paris_Biennale`.

3. Move the file in a PDF folder that you create in the corresponding `exhibCat_NAME_OF_THE_CATALOGUE` folder.

B.2.2 Non digitised Catalogues

If you need to process a non digitised catalogue :

1. Check with the institution if they have JPEG images of the pages or a PDF version of the catalogue and if they could give you a copy.

2. If they don't, you must scan or take photos of the pages. It is important to handle page by page and to pay attention to the quality of the photo (it should not be curve).

3. Create a folder for the catalogue named `exhibCat_NAME_OF_THE_CATALOGUE` in Catalogues folder. The name of the catalogue takes 3 arguments : the date, the location and the name of the exhibition. Here are some examples : `exhibCat_1892_Paris_SocieteArtistesIndependants`, `exhibCat_1921_TheHague_ExpositionHollandaise` and `exhibCat_1965_Paris_Biennale`.

4. When you have all your pages in JPEG or PDF, you can move them in a JPG or PDF folder that you create in the corresponding `exhibCat_NAME_OF_THE_CATALOGUE` folder.

B.3 Step 2 : OCR process on catalogues and export of the ALTO files

Now we need to process catalogues in a OCR software to get the text of the document. The transcription of each catalogue is augmented by typographical information : bold and italic words are tagged with HTML tags (``, ``, `<i>` and `</i>`). Then we will use two scripts written by Ljudmila Petkovic, a first one (1_ eval _txt⁴) to verify the good formation of the tag and another one (2_ ALTO_ XML_trans⁵) to correct automatically the malformed tags and to transform the ALTO-XML files come from the OCR software.

B.3.1 OCR process

Prerequisites

The following tutorial is using the Transkribus software. You can download it here⁶ and have access to differents guides.

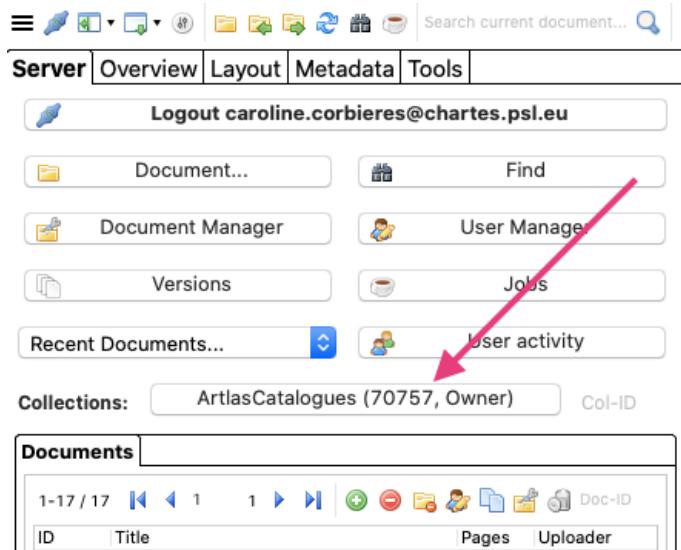
4. <https://github.com/carolinecorbieres/ArtlasCatalogues/tree/master/2_OCR/1_eval_txt>
 5. <https://github.com/carolinecorbieres/ArtlasCatalogues/tree/master/2_OCR/2_ALTO_XML_trans>
 6. <<https://transkribus.eu/Transkribus/>>

Run the OCR

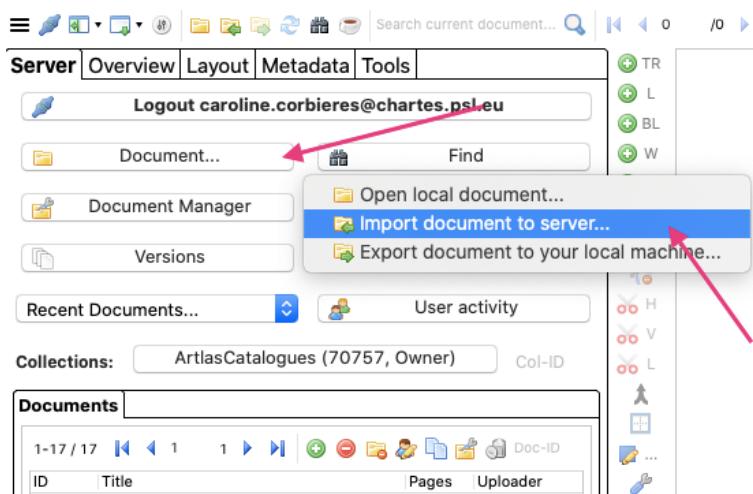
First you need to run the Transkribus app and to log in. If you don't have a Transkribus account, you can register on the website.

In Transkribus :

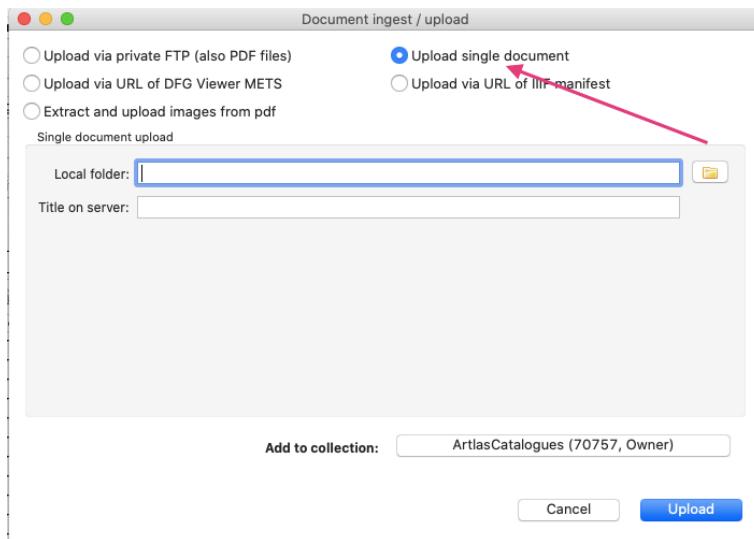
1. Select the **ArtlasCatalogues** collection (n° 70757). If you don't have access to it, you can contact Caroline Corbières at caroline.corbieres(at)chartes.psl.eu, she will give you an access to it.



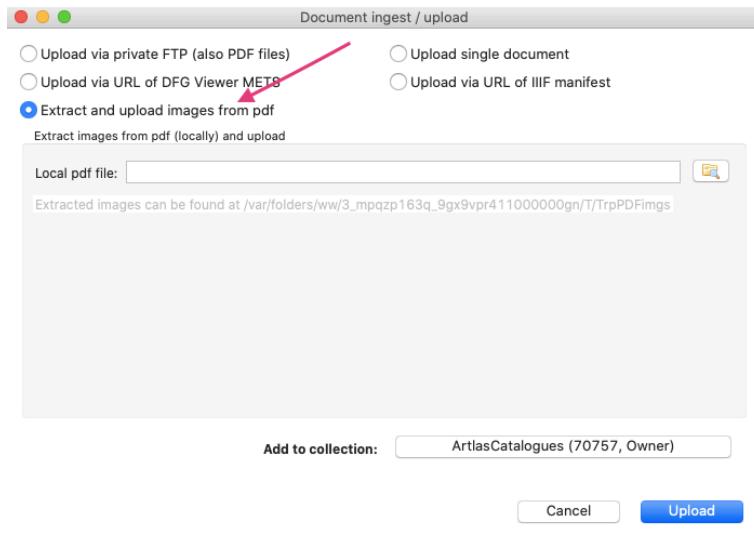
2. Import your catalogue in the collection. Go to **Document...** and click on **Import document to the server**.



- If you have JPEG images of your catalogue, choose **Upload single document** and add the path to the `exhibCat_NAME_OF_THE_CATALOGUE/JPG` folder you created before. You can add the name of the catalogue in the **Title on server** section.

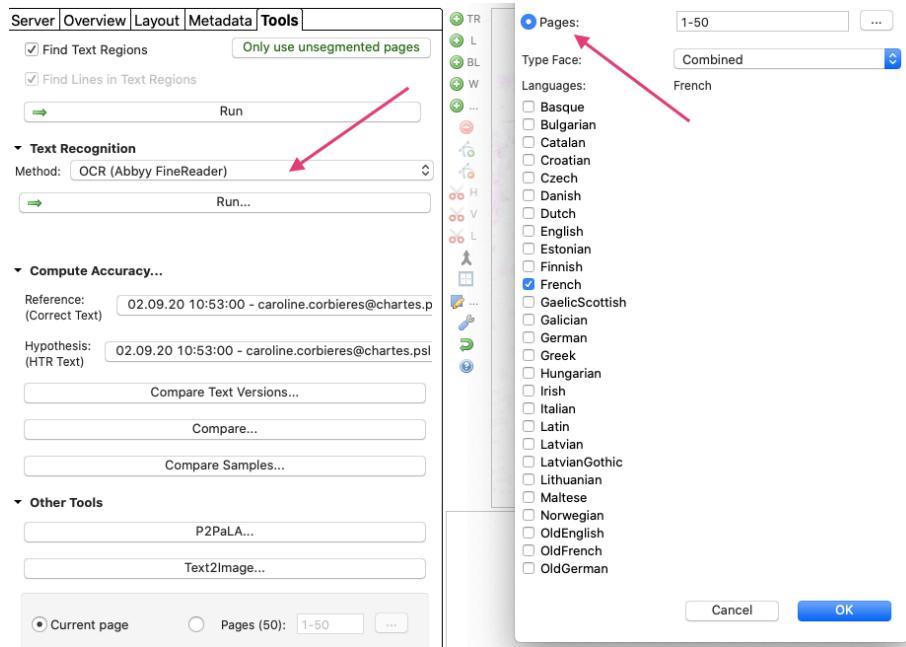


- If you have a PDF version of your catalogue, choose Extract and upload images from pdf and add the path to your exhibCat_NAME_OF_THE_CATALOGUE/PDF folder you created before. You can add the name of the catalogue in the Title on server section.



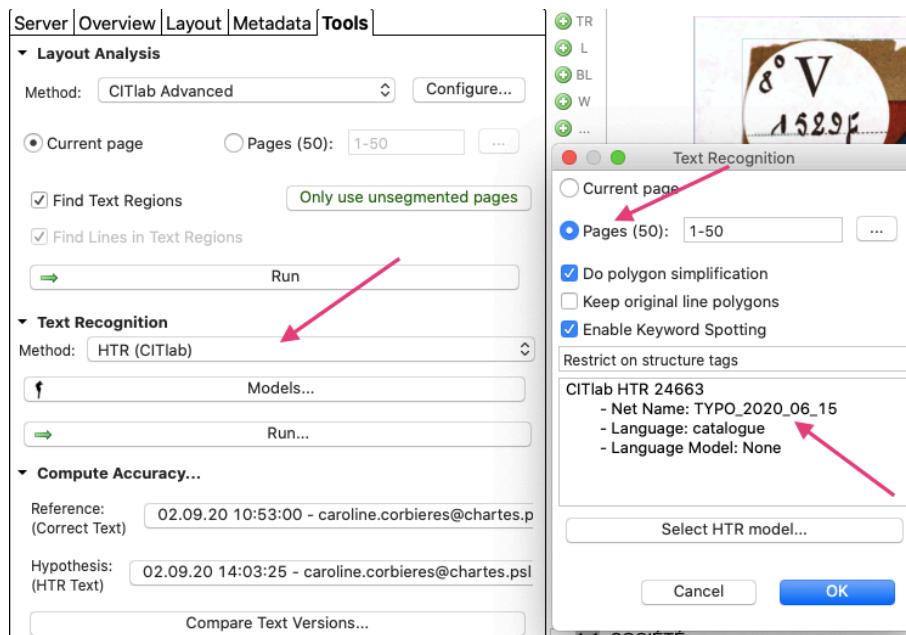
3. Run the OCR. Select your catalogue and go to the Tools tab.

- In Text Recognition select OCR (Abbyy FineReader) method and click on Run....
- Then select Pages in the new window and the language(s) of the catalogue and click on OK.



4. Run the TYPO_2020_06_15 HTR model. Go to the Tools tab.

- In Text Recognition select HTR (CITLab) method and click on Run....
- Then select Pages in the new window. If the TYPO_2020_06_15 HTR model is already stored, click on OK. If not, verify if you have access to it in the Select HTR model... tab, select it and click on Ok. If you don't find the model, you can contact Simon Gabay to have access to it at simon.gabay(at)unine.ch.



5. Verify the transcription of the OCR page by page. Go to the Layout tab.

- If you notice character strings which don't make sense and which aren't a transcription of the text, you can delete the line by selecting the TextRegion and click

on the no-entry sign. You can also delete some wrong characters of a line directly in the transcription area of Transkribus, under the image of the catalogue page.

The screenshot shows the Transkribus software interface. On the left is the 'Layout' tab of the 'Server Overview' window, displaying a hierarchical tree of text regions and lines. A red arrow points from the 'Delete selected shapes' button at the top of this tree to the transcription area on the right. The transcription area contains the text of a historical document, with several lines highlighted in blue or green. Below the transcription is a list of numbered entries corresponding to the lines in the tree.

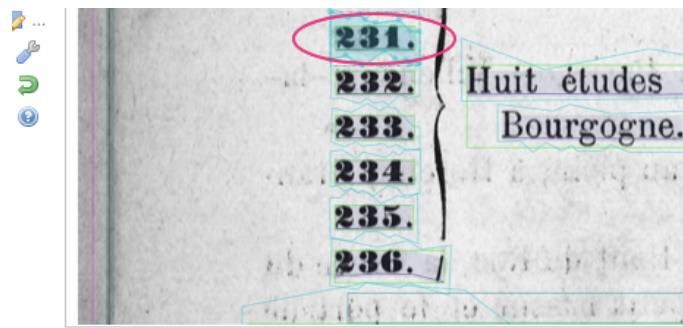
| Line Number | Transcription |
|-------------|---------------------------------|
| 1-1 | 1890 |
| 2-1 | 6. EXPOSITION |
| 3-1 | Pavillon de la Ville de Paris |
| 4-1 | (Champs-Élysées) |
| 5-1 | 68(14ήληχμιμπρτσαT 22328 |
| 6-1 | Ét ξz-h.? gl, auziere :Z44 |
| 7-1 | 123 (354.1 |
| 8-1 | .661 (2234-4. |
| 9-1 | 17. sizz (4ή3... |
| 10-1 | ...6317. zl, . i9cig'g< (5z2vlg |

- If you notice the following mistakes, you must correct them :
- A line isn't highlighted in blue and/or green : in the page image area, add a baseline (select the BL green button) under the line to create a TextRegion and write the corresponding transcription in the transcription area.

This screenshot shows the Transkribus interface with a historical document image. A red arrow points from the 'BL' button in the toolbar to the stamp in the bottom right corner of the image. The stamp contains text in French and English, including 'FONDATION' and 'SMITH-LESCUEF'. The transcription area below the image contains numbered entries corresponding to the lines in the layout editor. A red arrow also points from the bottom of the stamp area to the transcription area.

| Line Number | Transcription |
|-------------|---|
| 19-1 | *320. Le quai Henri IV. Paris. |
| 20-1 | *321. Notre-Dame. Paris. |
| 21-1 | *322. Saint-Michel d'Aiguilhe (Haute-Loire). |
| 22-1 | DULAC, Charles, né à Paris. — 32, rue |
| 23-1 | 323. Saint-Julien-le-Pauvre. Par |
| 24-2 | 324. Vézelay, cathédrale. |
| 25-1 | 325. Église, Pont-Aubert. |
| 26-1 | 326. Vézelay. |
| 27-1 | 327. Nature morte. |
| 28-1 | 328. Raies. |
| 29-1 | 329. Deux études marines. |

- A bold or italic word isn't between tags (you don't need to correct every tag, a script will do it) : encode the word with the corresponding tags in the transcription area.



17-2 227. Un portrait.
 17-3 228. Le pain à la ferme.
 18-1 229
 19-1 230.
20-1 231. ←
 21-1 232
 22-1 233.
 23-1 234.
 24-1 235b>

- Lines aren't in the good order : replace them in the Layout area.

| Type | Text | Structure | Reading | ID |
|------------|-----------|-----------|---------|----|
| Page | | | | |
| Printspace | | | | |
| TextRegion | paragraph | 1 r_1_1 | | |
| TextRegion | paragraph | 1 tl_1 | | |
| TextRegion | paragraph | 2 r_2_1 | | |
| TextRegion | paragraph | 1 tl_2 | | |
| TextRegion | paragraph | 3 r_2_2 | | |
| TextRegion | paragraph | 1 tl_3 | | |
| TextRegion | paragraph | 4 r_2_3 | | |
| TextRegion | paragraph | 1 tl_4 | | |
| TextRegion | paragraph | 5 r_2_4 | | |
| TextRegion | paragraph | 1 tl_5 | | |
| TextRegion | paragraph | 6 r_2_5 | | |
| TextRegion | paragraph | 1 tl_6 | | |
| TextRegion | paragraph | 7 r_3_1 | | |
| TextRegion | paragraph | 1 tl_7 | | |
| TextRegion | paragraph | 8 r_4_1 | | |
| TextRegion | paragraph | 1 tl_8 | | |
| TextRegion | paragraph | 2 tl_9 | | |
| TextRegion | paragraph | 9 r_4_2 | | |
| TextRegion | paragraph | 1 tl_10 | | |
| TextRegion | paragraph | 10 r_4_3 | | |
| TextRegion | paragraph | 1 tl_11 | | |
| TextRegion | paragraph | 2 tl_12 | | |
| TextRegion | paragraph | 11 r_4_4 | | |
| TextRegion | paragraph | 1 tl_13 | | |
| TextRegion | paragraph | 12 r_4_5 | | |
| TextRegion | paragraph | 1 tl_14 | | |
| TextRegion | paragraph | 13 r_4_6 | | |
| TextRegion | paragraph | 1 tl_15 | | |
| TextRegion | paragraph | 14 r_5_1 | | |
| TextRegion | paragraph | 15 r_6_1 | | |
| TextRegion | paragraph | 1 tl_17 | | |

— 13 —

*215. Tête d'étude. Pastel.
 *216. Un pré à Cornilla. (Assis dans le sillage).
 *217. Trocadéro. Effet du matin.
 *218. Souvenir de l'Exposition.
 *219. Souvenir de l'Exposition. Effet du soir.
 Pastel.

DANIEL, Georges, né à New-York. — 55, rue du Château (Paris).
 *220. Un mas à Cornilla de Conflent (Pyrénées-Orientales).
 *221. Une prairie.
 *222. Un pré à Cornilla (appartient à M. L. J. B.).
 *223. Leucate (Aude).
 *224. Le modèle.
 *225. Fleurs.
 *226. Un verger à Saint-Clément, près Cornilla.

DAVRIOT, Étouard, né à Beaune (Côte-d'Or). — 22, rue de la Tour-d'Auvray (Paris).
 *227. Un portrait.
 *228. Le pain à la ferme.
 *229. Huit études et impressions. Algérie et Bourgogne.
 *230. Le pont Rouge (le matin).
 *231. Le pont Rouge (le soir).

DAVIGNY, Joseph, né à Paris. — 76, rue de Passy (Paris).
 *232. Incident électoral.
 *233. Le pont Rouge (le matin).
 *234. Le pont Rouge (le soir).

10-1 222. Un pré à Cornilla (appartient à M. E.-J. B.).
 10-2 223. Leucate (Aude).
 11-1 224. Le modèle.
 12-1 225. Fleurs.
 13-1 226. Un verger à Saint-Clément, près Cornilla.
 14-1 Pastel.
 15-1 (Paris).

- A Table region in the Layout area : delete it and re-create baselines. We notice that the transcription of a TableCell isn't recorded in the ALTO files, so you must correct it.

The screenshot shows a comparison between a digital transcription table and a physical document page. On the left, a table lists transcription details for various lines, including line numbers, transcription, and corresponding line numbers in the original document. An arrow points from the top of the table towards the right side of the image. On the right, a scanned page from a catalog is shown with numbered entries (737, 738, 739, 740) and their descriptions.

| | | | |
|-------------------------------|-----------------|----|---------|
| > Line | <>De Cassis... | 1 | tl_28 |
| > Line | <>juin</i> 1... | 2 | tl_29 |
| ▼ Table | | 23 | tbl_3_3 |
| ▼ TableCell TrpTableCellIT... | paragraph | | r_3_1_1 |
| ► Line | | 1 | tl_30 |
| ▼ TableCell TrpTableCellIT... | paragraph | | r_3_1_2 |
| ► Line | b737 Op... | 1 | tl_31 |
| ▼ TableCell TrpTableCellIT... | paragraph | | r_3_2_1 |
| ► Line | | 1 | tl_32 |
| ▼ TableCell TrpTableCellIT... | paragraph | | r_3_2_2 |
| ► Line | 738b> Op. 1... | 1 | tl_33 |
| ▼ TableCell TrpTableCellIT... | paragraph | | r_3_3_1 |
| ► Line | | 1 | tl_34 |
| ▼ TableCell TrpTableCellIT... | paragraph | | r_3_3_2 |
| ► Line | b739 Op... | 1 | tl_35 |
| ▼ TableCell TrpTableCellIT... | paragraph | | r_3_4_1 |
| ► Line | | 1 | tl_36 |
| ▼ TableCell TrpTableCellIT... | paragraph | | r_3_4_2 |
| ► Line | | 1 | tl_37 |
| ▼ TableCell TrpTableCellIT... | paragraph | | r_3_5_1 |
| ► Line | | 1 | tl_38 |
| ▼ TableCell TrpTableCellIT... | paragraph | | r_3_5_2 |
| ► Line | 740 Op.... | 1 | tl_39 |

1-1 — 36
2-1 718. Faunesse.
3-1 719. La tâche.
4-1 720. Retour d'école.
5-1 721. Misère noire.
6-1 722. Petite vendange.
7-1 SERVAL, Maurice, né à Douai, — 26, rue Bréda (Paris).
7-2 723. Intérieur. Aquarelle.

Don't forget to save the transcription after each correction.

B.3.2 Tag verification and correction

1. In Transkribus, export the transcription in .txt. Go to the folder logo with a green right arrow.

- In Client export tab, file the path to your folder : YOUR_PATH_TO_THE_FOLDER/visual-contagions/Catalogues/2_0CR/1_eval_txt/doc and the file name : exhibCat_NAME_OF_THE_CATALOGUE.
- In Choose export formats, deselect Transkribus Document and select Simple TXT.
- Click on OK.
- Go to the folder you just created and move the exhibCat_NAME_OF_THE_CATALOGUE.txt in the doc folder. You can delete the exhibCat_NAME_OF_THE_CATALOGUE folder created by Transkribus.

2. Go to your visual-contagions/Catalogues/2_0CR/1_eval_txt/script folder and open the score_corr.py script in a text editor program. - Complete the path to your document in the line 9 : ../doc/exhibCat_NAME_OF_THE_CATALOGUE.txt.

```

1 import re
2 import sys
3 import lxml.etree as etree
4 import os
5
6 ##### Processing the files from the command line #####
7
8 # !! COMPLETE THE NAME OF THE CATALOGUE !!
9 fichier = '../doc/exhibCat_NAME_OF_THE_CATALOGUE.txt' ←
10
11 ##### Regexes #####
12

```

3. Run in the terminal the following commands : - Go to the script folder.

```
cd YOUR_PATH_TO_THE_FOLDER/visual-contagions/Catalogues/2_0CR/1_eval_txt/script
```

- Run the python script.

```
python3 score_corr.py
```

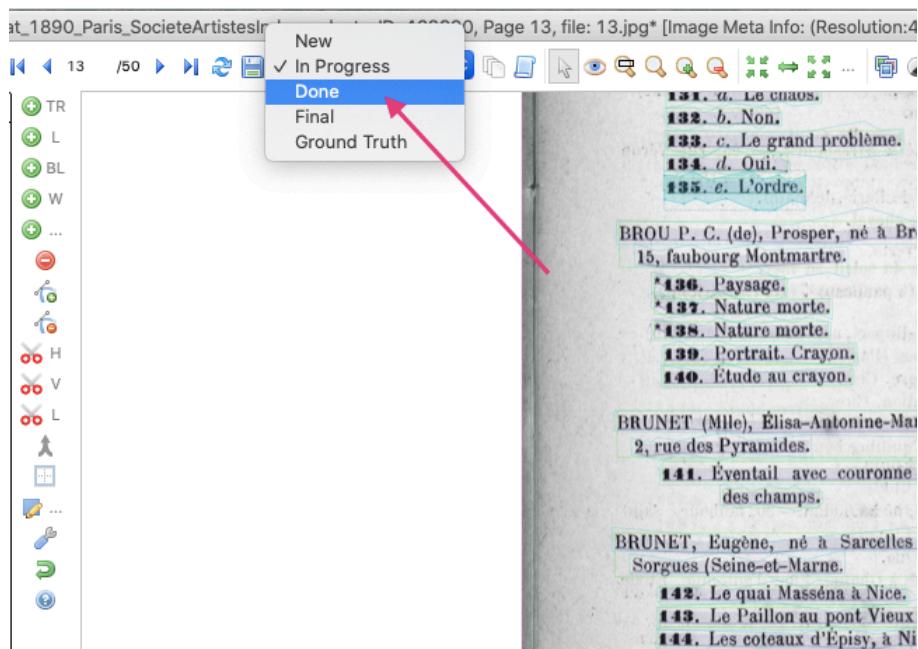
The script output is a the table of 3 columns : - The first one corresponds to the original text.

- The second one indicates a number corresponding to the possible tag scenarios. You can find more informations about it here⁷.
- The third one could be either the error signalisation or the suggestion of the correction.

4. In the second column of the script output, find the 2, 3 and 4 numbers to correct the OCR.

| | | |
|---|---|--------------------------|
| - 14 - | | |
| *240. Sentier à Villejuif. | 0 | |
| *241. Entrée de village. | 1 | |
| 242. Il neige. | 1 | BALISES MANQUANTES |
| *243. La dives. | 1 | |
| >*244. A Villejuif. | 1 | *244 A Villejuif. |
| | 1 | |
| *245. Deux belles vieillesses. | 0 | |
| DEBRAY, Eugène-Frédéric, né à Paris. - 47, rue des Batignolles. | 0 | |
| | 0 | |

- Return in Transkribus and correct the malformed tags (visible thanks to the numbers 2, 3 or 4) in the transcription area. When you have corrected a page, you can update the **In Progress** status (above the page image) to **Done** status. At the end, all your pages must be Done (you can check it in the **Overview** tab).

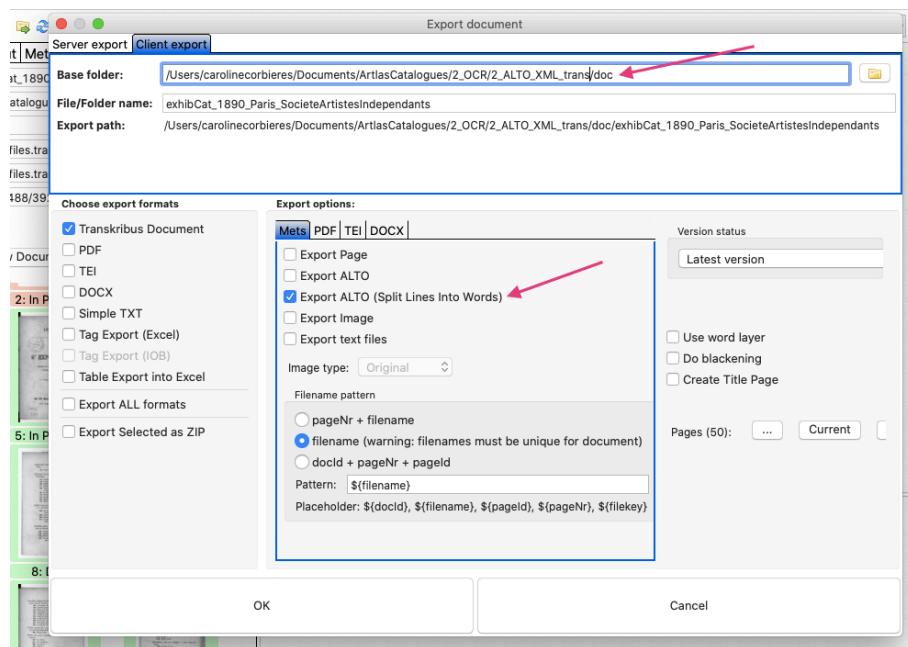


7. <https://github.com/ljpetkovic/OCR-cat/tree/GROBID_eval/eval_txt>

B.3.3 Transformation of the ALTO-XML files

1. In Transkribus, export the transcription in ALTO. Go to the folder logo with a green right arrow.

- In Client export tab, file the path to your folder : YOUR_PATH_TO_THE_FOLDER/visual-contagions/Catalogues/2_0CR/2_ALTO_XML_trans/doc.
- In Export options, deselect Export Page and Export Image and select Export ALTO (Split Lines Into Words).
- Click on OK.



— Go to the folder you just created and move the ALTO files (which are in the `alto` folder) in the `doc/exhibCat_NAME_OF_THE_CATALOGUE` folder. You can delete the `alo` folder and the `.xml` files created by Transkribus.

2. To transform the ALTO files and correct the OCR, run in the terminal the following commands :

- Go to the `script` folder.

```
cd YOUR_PATH_TO_THE_FOLDER/visual-contagions/Catalogues/2_0CR/2_ALTO_
XML_scripts
```

- Run the `.sh` script.

```
bash corr_trans_ALTO.sh -d exhibCat_NAME_OF_THE_CATALOGUE
```

3. Move the `_trans.xml` files created in the `scripts` folder to the `Catalogues/exhibCat_NAME_OF_THE_CATALOGUE/ALTO` folder. In order to use GROBID, you can delete all ALTO pages that don't record catalogue entries.

B.3.4 Credits

The python scripts are written by Ljudmila Petkovic, with the help of Simon Gabay, you can find more informations about them on her GitHub⁸.

B.4 Step 3 : Transformation of the ALTO files to PDF files

While we are waiting for GROBID to process ALTO files, we add to the workflow this intermediate step to transform ALTO files to PDF files. The `alto_to_fo.xsl` xsl stylesheet transforms the ALTO files to XML-FO files by keeping bold and italic informations. Then a FO processor (like XEP) transforms the XML-FO files to PDF files.

B.4.1 Prerequisites

The following tutorial is using Oxygen XML Editor⁹ and RenderX XEP Engine¹⁰.

Oxygen

You can download Oxygen XML Editor here¹¹.

RenderX

You can have a RenderX free personal (or academic) licence here¹².

After you have installed RenderX on your computer, you need to add the XEP processor to Oxygen. In Oxygen, go to :

`Options > Préférences... > XML > Sortie PDF > Processeurs FO` (arrow 1) and click on `Parcourir` (arrow 2) next to "Ajouter un processeur FO 'XEP' (un fichier exécutable est nécessaire)".

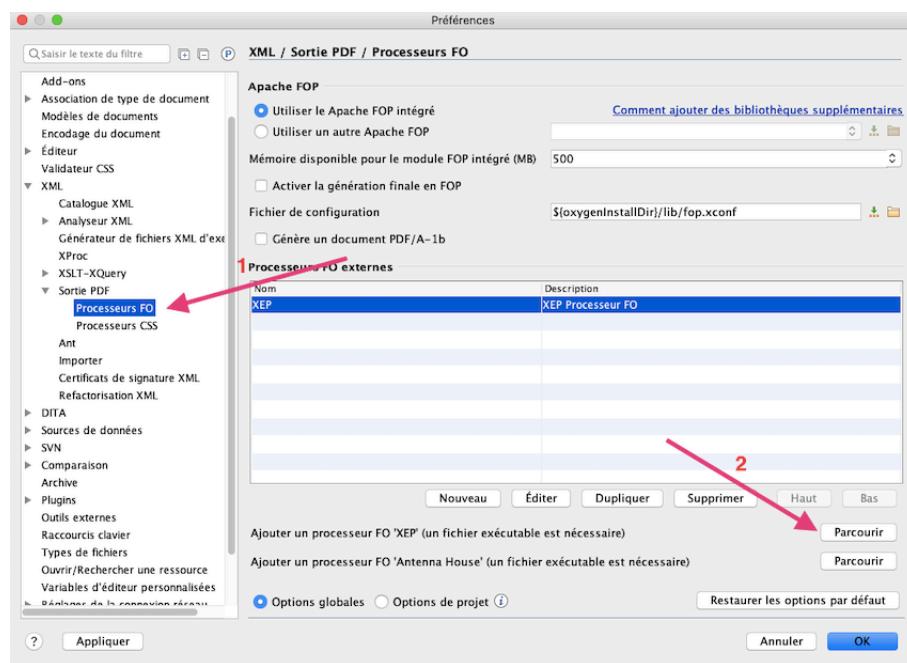
8. <<https://github.com/ljpetkovic/OCR-cat>>

9. <<https://www.oxygenxml.com/>>

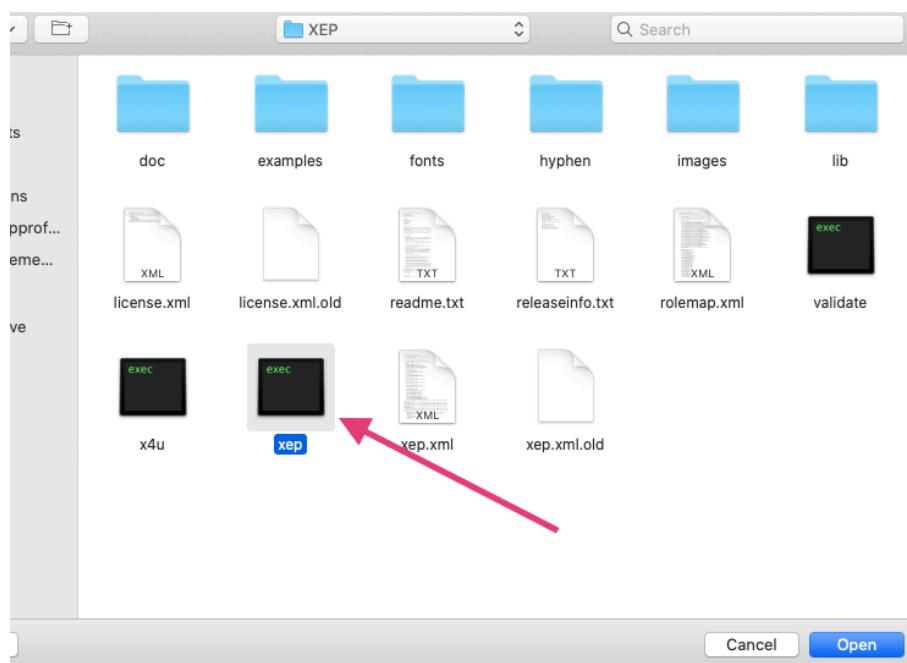
10. <<http://www.renderx.com/tools/xep.html>>

11. <https://www.oxygenxml.com/xml_editor/download_oxygenxml_editor.html>

12. <<http://www.renderx.com/download/personal.html>>



Then add the path to your `xep` script, in the `XEP` folder you have downloaded.

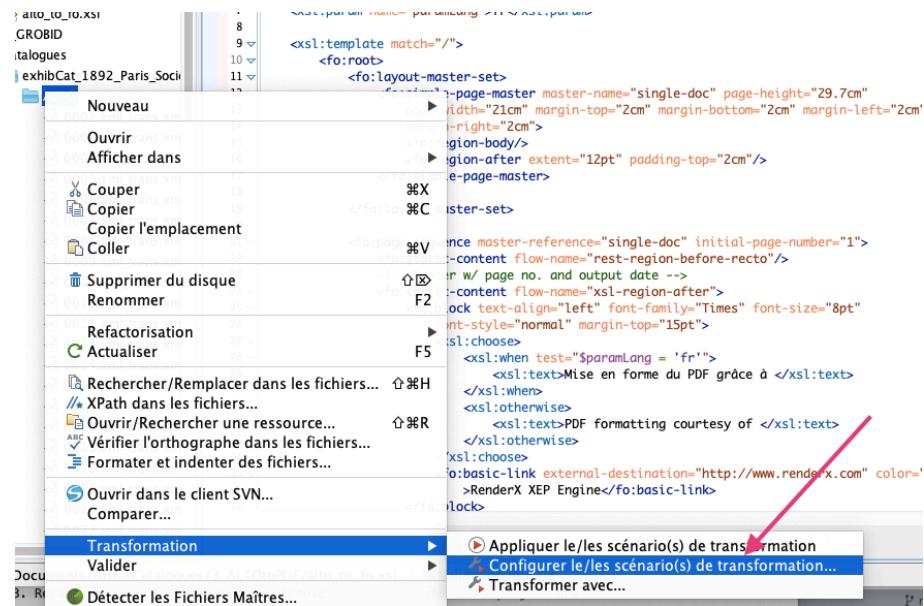


B.4.2 Transformation

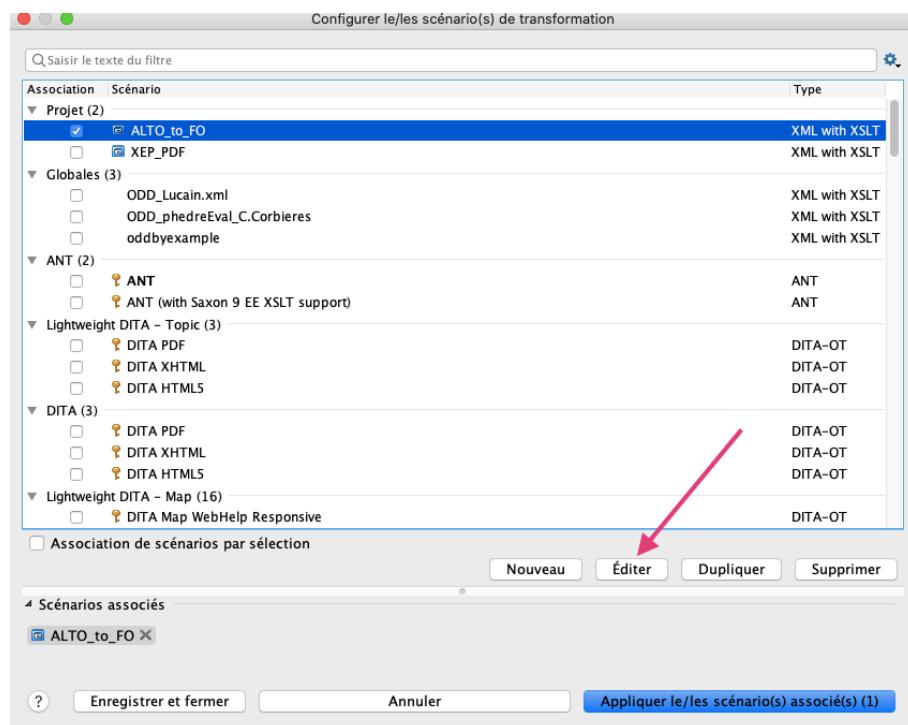
First, you need to open `ExhibitionCatalogues.xpr` project in Oxygen.

ALTO to FO

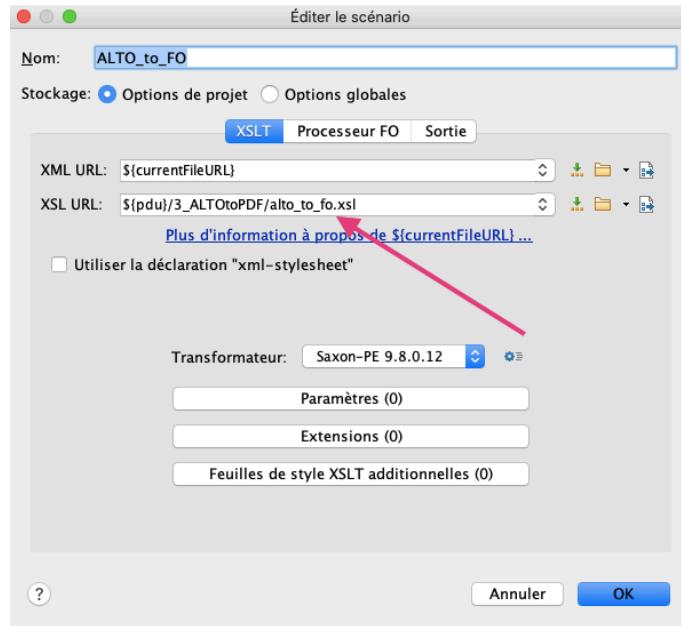
1. Go to Catalogues > `exhibCat_NAME_OF_THE_CATALOGUE` > ALTO and click on Transformation > Configurer le/les scénario(s) de transformation....



2. Select the ALTO_to_FO scenario and click on Éditer.

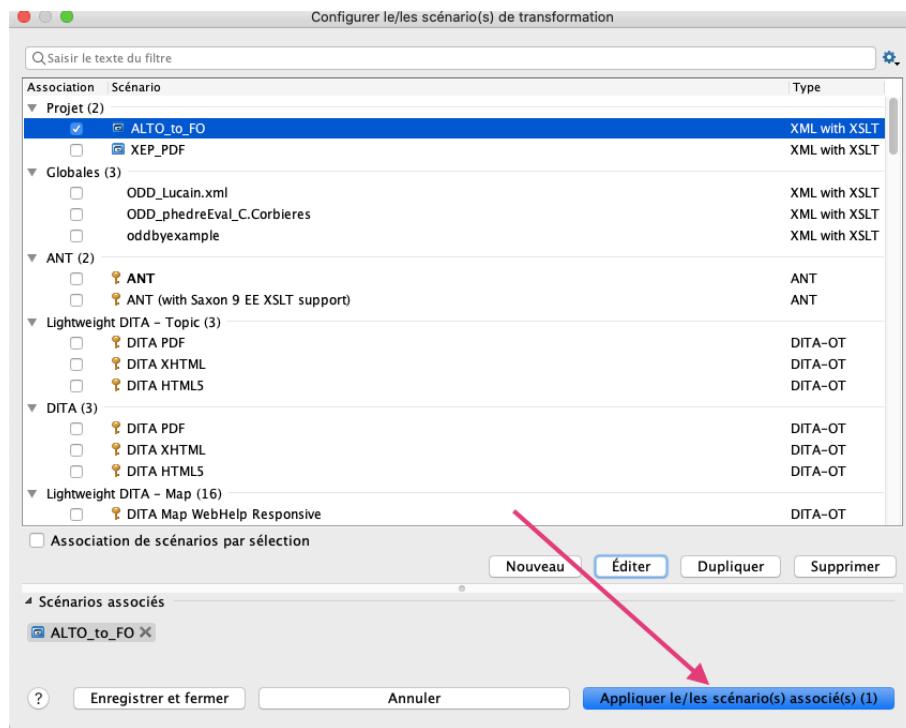


3. Check the XSL URL, it should match with your path on the alto_to_fo.xsl xsl stylesheet.



4. Click on OK.

5. Then select the ALTO_to_FO scenario and click on Appliquer le/les scénario(s) associé(s).

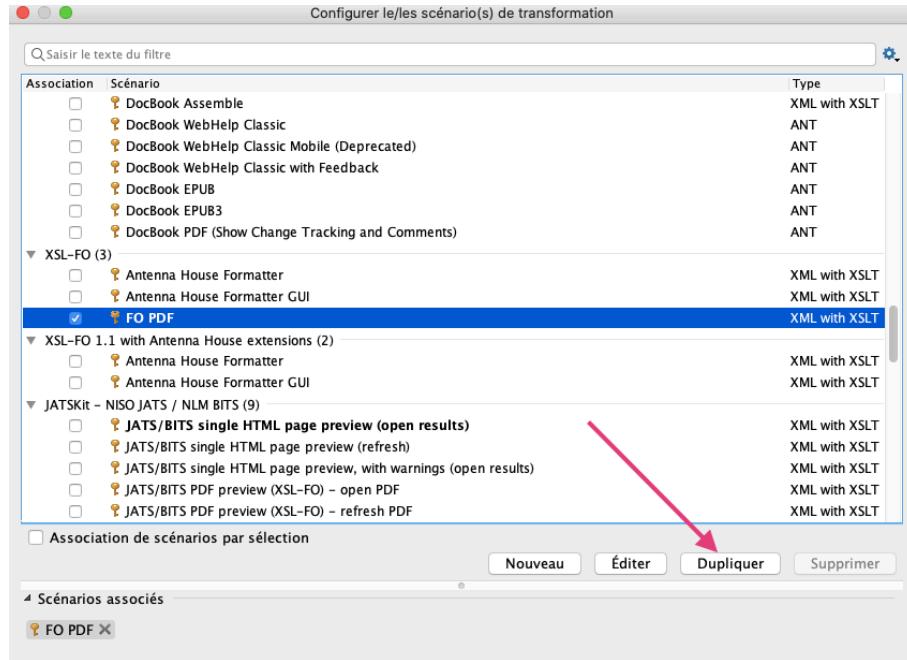


6. Move the FO files created to F0 folder.

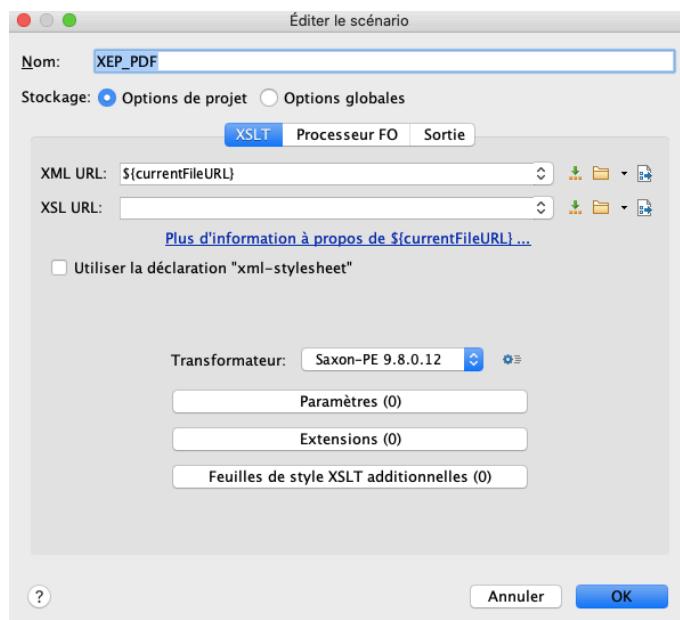
FO to PDF

Now, you need to create a transformation scenario using XEP processor (you just have to do this step once) :

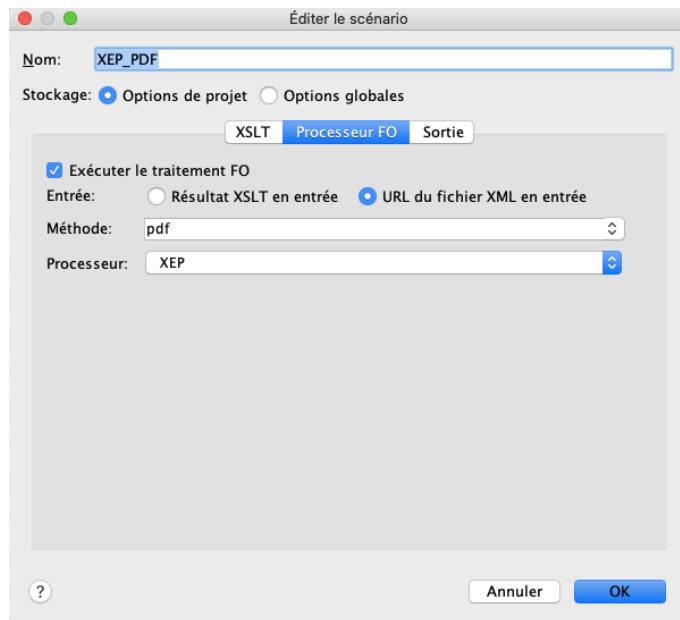
1. Go to Catalogues > exhibCat_NAME_OF_THE_CATALOGUE > FO and click on Transformation > Configurer le/les scénario(s) de transformation....
2. Duplicate the FO PDF scenario.



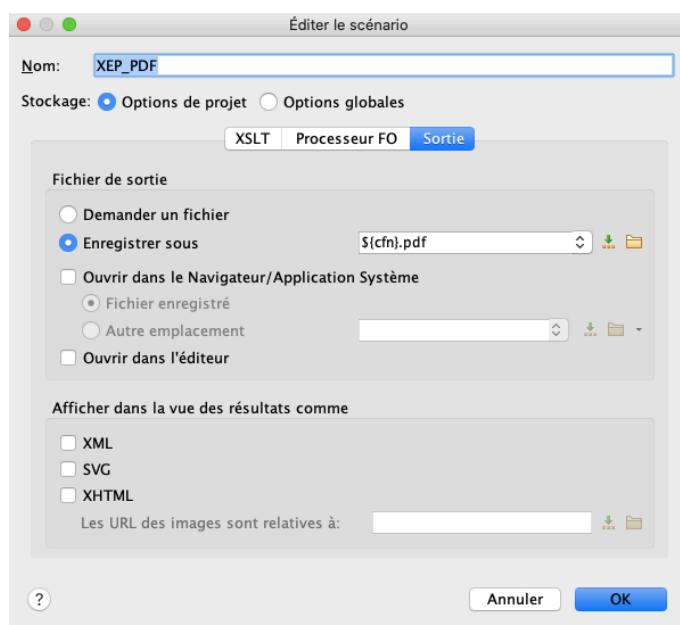
3. Rename it XEP_PDF for example, choose Stockage: Options de projet and Transformateur: Saxon-PE 9.8.0.12.



4. In Processeur FO tab, click on exécuter le traitement FO and select Méthode: pdf and Processeur: XEP.



5. In Sortie tab, select Enregistrer sous and write : \${cfn}.pdf.



6. Click on OK.

7. Then select the XEP_PDF scenario and click on Appliquer le/les scénario(s) associé(s).

8. Move the PDF files created to PDF folder.

Once you have created the XEP_PDF scenario, you can skip steps 2 to 6.

Put PDF files together with cpdf

Finally you need to put all your PDF pages together by using cpdf (Coherent PDF Command Line Tools) tool. In order to use GROBID, you can delete all PDF pages that don't record catalogue entries.

1. Download `cpdf` tool here¹³.

2. If you are on Mac OS, you first need to remove the `.DS_Store` file of your PDF folder. Run in your terminal the following commands :

- Go to your folder.

```
cd YOUR_PATH_TO_THE_FOLDER/visual-contagions/Catalogues/Catalogues/
exhibCat_NAME_OF_THE_CATALOGUE/PDF/
```

- Look for hidden files. `ls -a`

- If there is a `.DS_Store` file, remove it. `rm .DS_Store`

- Check if you deleted the `.DS_Store` file. `ls -a`

3. Use `cpdf` tool to create a new PDF. Run in your terminal the following commands :

- Activate the `cpdf` tool.

```
cd YOUR_PATH_TO_THE_FOLDER/cpdf-binaries-master/OSX-Intel/
```

- Put all your PDF pages together.

```
./cpdf -merge -idir ~/YOUR_PATH_TO_THE_FOLDER/visual-contagions/Catalogues
/Catalogues/exhibCat_NAME_OF_THE_CATALOGUE/PDF/ -o ~/YOUR_PATH_TO_THE_
FOLDER/ArtlasCatalogues/Catalogues/exhibCat_NAME_OF_THE_CATALOGUE/PDF/
exhibCat_NAME_OF_THE_CATALOGUE-FO.pdf
```

B.4.3 Thanks

Thanks to Jean-Paul Rehr who explained us how to use XML-FO and XEP processor.

B.5 Step 4 : Using GROBID-dictionaries to automatically encode exhibition catalogues

GROBID-dictionaries is a machine learning software for structuring in XML-TEI digitised documents like lexical ressources, dictionaries and bibliographies. We also use it to encode in XML-TEI sale catalogues¹⁴ and exhibition catalogues.

B.5.1 Choose the right train set

To encode a catalogue, you need to choose the train set that matches your catalogue. For now, we only have few train sets but you can train some more and add them here.

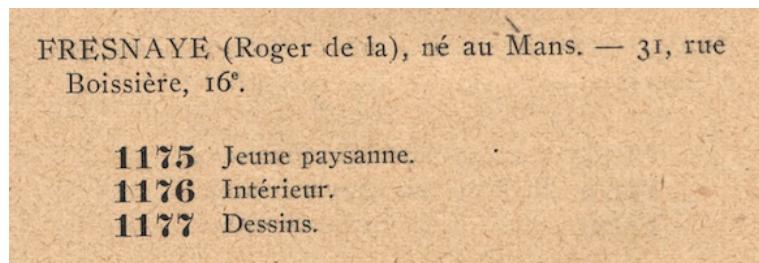
13. <<https://community.coherentpdf.com/>>

14. <https://github.com/katabase/GROBID_Dictionaries>

When to choose the `trainingData_INDEPENDANTS` train set ?

If the entries of your exhibition catalogue are the same as the *Catalogue de la société des artistes indépendants*, you can choose this train set for using GROBID-dictionaries. You need to pay attention to following typographical details :

- the exhibitor name is in capital ;
- his address is separated from the rest of the biographical informations by an underscore ;
- the number of each exhibited work is in bold

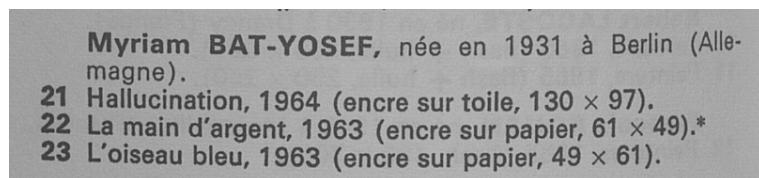


The train set contains extracted data from several catalogues of the *Société des artistes indépendants* (1890, 1892, 1913, 1923).

When to choose the `trainingData BIENNALE` train set ?

If the entries of your exhibition catalogue are the same as the *Catalogue de la Biennale de Paris*, you can choose this train set for using GROBID-dictionaries. You need to pay attention to following typographical details :

- the exhibitor name is in bold ;
- the number of each exhibited work is in bold ;
- some informations about the work are in parentheses.



The train set contains extracted data from several catalogues of the *Biennale de Paris* (1961, 1965) and of the *Bienal di Museu de Arte Moderna de Sao Paulo* (1951, 1972).

B.5.2 Encode a catalogue

Prerequisites

To use GROBID-dictionaries, you need to install Docker. You can download it here¹⁵.

Then you need to download GROBID-dictionaries. Run in your terminal the following command :

```
docker pull medkhem/grobid-dictionaries-stable
```

Run the train set

To run the trainings, you need to run in your terminal the following commands :

1. Synchronise the right train set.

```
docker run -v PATH_TO_YOUR_FOLDER/visual-contagions/Catalogues/4_GROBID/
           trainingData_NAME_OF_THE_DATASET/toyData:/grobid/grobid-dictionaries/
           resources -p 8080:8080 -it medkhem/grobid-dictionaries-stable bash
```

Now, your terminal must start with something like that root@ef0ec5b02e82:/grobid/grobid-dictionaries#.

2. Run the learning process, one after another.

- Dictionary segmentation :

```
mvn generate-resources -P train_dictionary_segmentation -e
```

- Dictionary body segmentation :

```
mvn generate-resources -P train_dictionary_body_segmentation -e
```

- Lexical entry :

```
mvn generate-resources -P train_lexicalEntries -e
```

- Sense :

```
mvn generate-resources -P train_form -e
```

- Form :

```
mvn generate-resources -P train_sense -e
```

Now you trained all the levels of the model, you can run the web app to encode your catalogue.

Run the web app

1. Run in the terminal the following command :

```
mvn -Dmaven.test.skip=true jetty:run-war
```

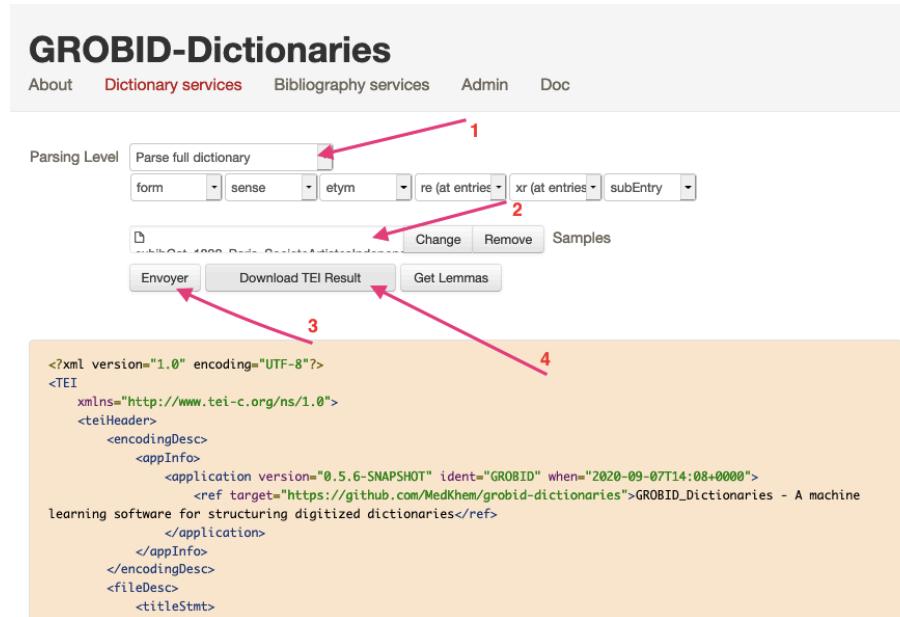
2. When you read [INFO] Started Jetty Server in your terminal, go to the localhost : http://localhost:8080/¹⁶

3. Upload your catalogue.

15. <<https://www.docker.com/>>

16. <<http://localhost:8080/>>

- In the Dictionary services tab, select Parse full dictionary.
- Upload your catalogue (in PDF).
- Click on Envoyer.
- Click on Download TEI Result.



4. You can leave the localhost by typing in the terminal `ctrl + C`.
5. Move the new TEI file in the `Catalogues/exhibCat_NAME_OF_THE_CATALOGUE/TEI` folder.

B.5.3 Create and train a new model

You can find here¹⁷ a guide to create new models and train existing ones in GROBID-dictionaries.

B.5.4 Credits and licence

GROBID-dictionaries is developed by Mohamed Khemakhem, you can find more informations about his work in his GitHub¹⁸. You also could find more informations on GROBID technologies here¹⁹.

You can find more informations about the licence of GROBID-dictionaries here²⁰.

17. <https://github.com/carolinecorbieres/ArtlasCatalogues/blob/master/4_GROBID/How_to_train_a_model.md>

18. <<https://github.com/MedKhem/grobid-dictionaries>>

19. <<https://grobid.readthedocs.io/en/latest/>>

20. <<https://github.com/MedKhem/grobid-dictionaries>>

B.6 Step 5 : Transformation of the .xml file automatically encoded by GROBID-dictionaries

GROBID-dictionaries offers an encoding for dictionaries. To have an encoding suitable for exhibition catalogues, we have to change some tags and add informations, like the header of the document. To do that, we use two xsl stylesheets.

B.6.1 Transformation 1

The `transformGROBIDoutput1` xsl stylesheet change the tags and allow us to correct the first mistakes of the encoding.

1. Open the catalogue encoded by GROBID-dictionaries and the `ExhibitionCatalogues.xpr` project in Oxygen.
2. Open the `transformGROBIDoutput1` xsl stylesheet and fill the catalogue's name.

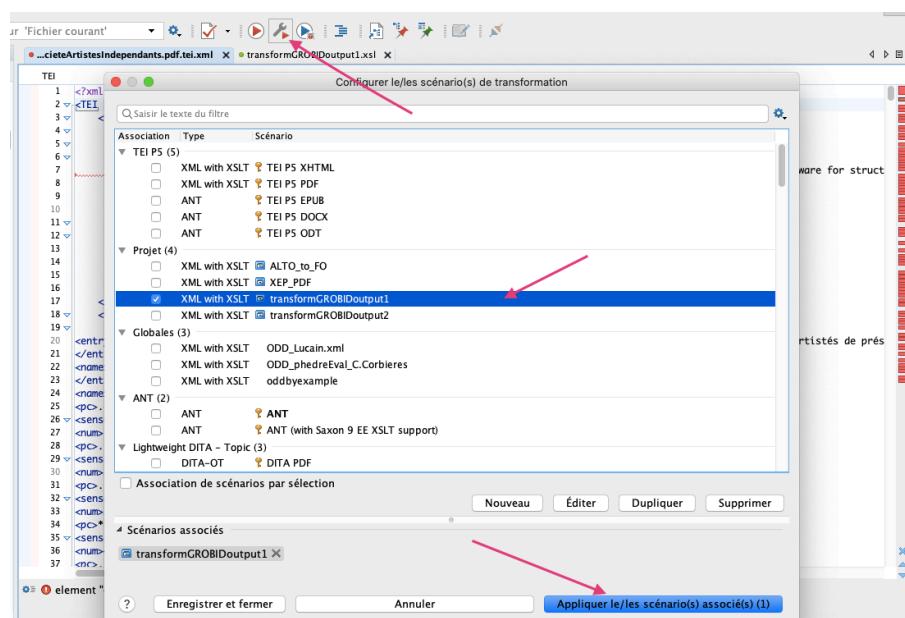


```

<xsl:stylesheet>
  <xsl:template match="></xsl:template>
    <xsl:attribute name="xml:id">exhibCat_NAME_OF_THE_CATALOGUE</xsl:attribute>
  </xsl:template>
</xsl:stylesheet>

```

3. Click on the red monkey wrench and select in Projet (4) : XML with XSLT `transformGROBIDoutput1`. Then click on Appliquer le/les scénario(s) associé(s).



it will open a new .xml document in Oxygen called exhibCat_NAME_OF_THE_CATALOGUE.pdf.tei_body.xml.

4. Correct the document until it is green.

- First, you have to correct the <application> tag like the following picture.



```

TEI text body
1  <?xml version="1.0" encoding="UTF-8"?>
2  <?xml-model href="../../_5_ImproveGROBIDoutput/ODD/ODD_Transformation.rng" type="application/xml" schematypens="http://relaxng.org/ns/structure/1.0"
3  <?xml-model href="../../_5_ImproveGROBIDoutput/ODD/ODD_Transformation.rng" type="application/xml" schematypens="http://purl.oclc.org/dsdl/schemas/tei-c/1.0" id="exhibCat_1892_Paris_SocieteArtistesIndependants">
4  <TEI xmlns="http://www.tei-c.org/ns/1.0"
5    id="exhibCat_1892_Paris_SocieteArtistesIndependants">
6    <teiHeader>
7      <encodingDesc>
8        <appInfo>
9          <application version="0.5.6" ident="GROBID" when="2020-09-08">
10         <ref target="https://github.com/MedKhem/grobid-dictionaries">GROBID_Dictionaries - A machine learning software for structuring
11         digitized dictionaries</ref>
12       </application>
13     </appInfo>
14   </encodingDesc>
15   <fileDesc>
16     <titleStmt>
17       <title level="a" type="main">
18         <titlestmt>
19       </titlestmt>
20     </fileDesc>
21   </teiHeader>
22 </text>

```

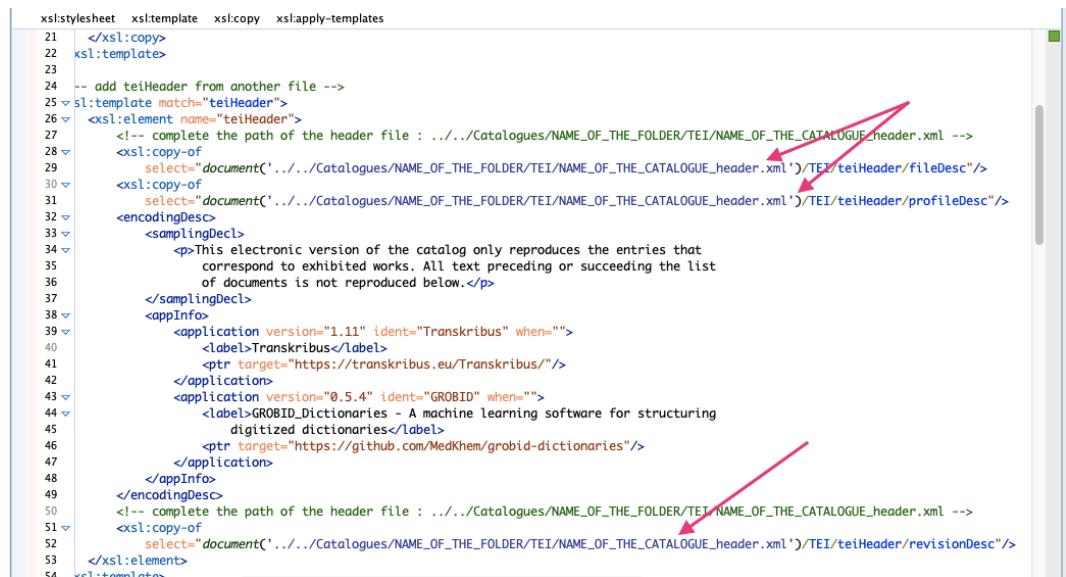
- Then you have to correct all the mistakes in the catalogue entries.

B.6.2 Transformation 2

The transformGROBIDoutput2 xsl stylesheet add the header and the xml:id in the file and allow us to correct the last mistakes of the encoding.

1. Open the transformGROBIDoutput2 xsl stylesheet.

- Fill the catalogue's name.
- Fill the catalogue's header path thrice.

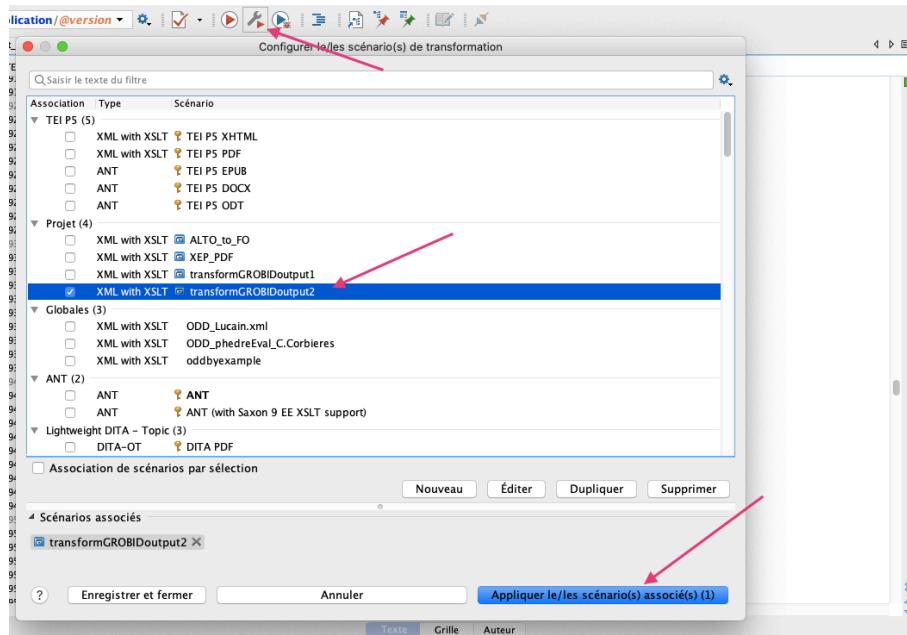


```

xslstylesheet xsl:template xsl:copy xsl:apply-templates
21  </xsl:copy>
22  <xsl:template>
23
24  -- add teiHeader from another file -->
25  <xsl:template match="teiHeader">
26    <xsl:element name="teiHeader">
27      <!-- complete the path of the header file : ../../Catalogues/NAME_OF_THE_FOLDER/TEI/NAME_OF_THE_CATALOGUE_header.xml -->
28      <xsl:copy-of select="document('../../Catalogues/NAME_OF_THE_FOLDER/TEI/NAME_OF_THE_CATALOGUE_header.xml')/TEI/teiHeader/fileDesc"/>
29      <xsl:copy-of select="document('../../Catalogues/NAME_OF_THE_FOLDER/TEI/NAME_OF_THE_CATALOGUE_header.xml')/TEI/teiHeader/profileDesc"/>
30    </xsl:copy-of>
31    <xsl:copy-of select="document('../../Catalogues/NAME_OF_THE_FOLDER/TEI/NAME_OF_THE_CATALOGUE_header.xml')/TEI/teiHeader/revisionDesc"/>
32  </xsl:copy-of>
33  <xsl:copy-of select="document('../../Catalogues/NAME_OF_THE_FOLDER/TEI/NAME_OF_THE_CATALOGUE_header.xml')/TEI/teiHeader/revisionDesc"/>
34  <xsl:copy-of select="document('../../Catalogues/NAME_OF_THE_FOLDER/TEI/NAME_OF_THE_CATALOGUE_header.xml')/TEI/teiHeader/revisionDesc"/>
35  <xsl:copy-of select="document('../../Catalogues/NAME_OF_THE_FOLDER/TEI/NAME_OF_THE_CATALOGUE_header.xml')/TEI/teiHeader/revisionDesc"/>
36  <xsl:copy-of select="document('../../Catalogues/NAME_OF_THE_FOLDER/TEI/NAME_OF_THE_CATALOGUE_header.xml')/TEI/teiHeader/revisionDesc"/>
37  <xsl:copy-of select="document('../../Catalogues/NAME_OF_THE_FOLDER/TEI/NAME_OF_THE_CATALOGUE_header.xml')/TEI/teiHeader/revisionDesc"/>
38  <xsl:copy-of select="document('../../Catalogues/NAME_OF_THE_FOLDER/TEI/NAME_OF_THE_CATALOGUE_header.xml')/TEI/teiHeader/revisionDesc"/>
39  <xsl:copy-of select="document('../../Catalogues/NAME_OF_THE_FOLDER/TEI/NAME_OF_THE_CATALOGUE_header.xml')/TEI/teiHeader/revisionDesc"/>
40  <xsl:copy-of select="document('../../Catalogues/NAME_OF_THE_FOLDER/TEI/NAME_OF_THE_CATALOGUE_header.xml')/TEI/teiHeader/revisionDesc"/>
41  <xsl:copy-of select="document('../../Catalogues/NAME_OF_THE_FOLDER/TEI/NAME_OF_THE_CATALOGUE_header.xml')/TEI/teiHeader/revisionDesc"/>
42  <xsl:copy-of select="document('../../Catalogues/NAME_OF_THE_FOLDER/TEI/NAME_OF_THE_CATALOGUE_header.xml')/TEI/teiHeader/revisionDesc"/>
43  <xsl:copy-of select="document('../../Catalogues/NAME_OF_THE_FOLDER/TEI/NAME_OF_THE_CATALOGUE_header.xml')/TEI/teiHeader/revisionDesc"/>
44  <xsl:copy-of select="document('../../Catalogues/NAME_OF_THE_FOLDER/TEI/NAME_OF_THE_CATALOGUE_header.xml')/TEI/teiHeader/revisionDesc"/>
45  <xsl:copy-of select="document('../../Catalogues/NAME_OF_THE_FOLDER/TEI/NAME_OF_THE_CATALOGUE_header.xml')/TEI/teiHeader/revisionDesc"/>
46  <xsl:copy-of select="document('../../Catalogues/NAME_OF_THE_FOLDER/TEI/NAME_OF_THE_CATALOGUE_header.xml')/TEI/teiHeader/revisionDesc"/>
47  <xsl:copy-of select="document('../../Catalogues/NAME_OF_THE_FOLDER/TEI/NAME_OF_THE_CATALOGUE_header.xml')/TEI/teiHeader/revisionDesc"/>
48  <xsl:copy-of select="document('../../Catalogues/NAME_OF_THE_FOLDER/TEI/NAME_OF_THE_CATALOGUE_header.xml')/TEI/teiHeader/revisionDesc"/>
49  <xsl:copy-of select="document('../../Catalogues/NAME_OF_THE_FOLDER/TEI/NAME_OF_THE_CATALOGUE_header.xml')/TEI/teiHeader/revisionDesc"/>
50  <xsl:copy-of select="document('../../Catalogues/NAME_OF_THE_FOLDER/TEI/NAME_OF_THE_CATALOGUE_header.xml')/TEI/teiHeader/revisionDesc"/>
51  <xsl:copy-of select="document('../../Catalogues/NAME_OF_THE_FOLDER/TEI/NAME_OF_THE_CATALOGUE_header.xml')/TEI/teiHeader/revisionDesc"/>
52  <xsl:copy-of select="document('../../Catalogues/NAME_OF_THE_FOLDER/TEI/NAME_OF_THE_CATALOGUE_header.xml')/TEI/teiHeader/revisionDesc"/>
53  <xsl:copy-of select="document('../../Catalogues/NAME_OF_THE_FOLDER/TEI/NAME_OF_THE_CATALOGUE_header.xml')/TEI/teiHeader/revisionDesc"/>
54  <xsl:element>

```

2. Click on the red monkey wrench and select in Projet (4) : XML with XSLT transformGROBIDoutput2. Then click on Appliquer le/les scénario(s) associé(s).



It will open a new .xml document in Oxygen called exhibCat_NAME_OF_THE_CATALOGUE.pdf.
tei_body_step2.xml.

3. Fill the header : enter your name in the <editor role="data"> tag.

```

TEI teiHeader fileDesc publicationStmt publisher orgName
3 <xml-model href="http://www.tei-c.org/ns/1.0">
4 <TEI xmlns="http://www.tei-c.org/ns/1.0">
5   <teiHeader>
6     <fileDesc>
7       <titleStmt>
8         <title>Société des artistes indépendants. Catalogue des œuvres exposées.</title>
9         <editor role="metadata">
10          <persName>Auriane Quoix</persName>
11        </editor>
12        <editor role="data">
13          <persName><!-- Prénom et nom de la personne ayant encodé le corps du texte --></persName>
14        </editor>
15       </titleStmt>
16     <publicationStmt>
17       <publisher>
18         <name>VISUAL CONTAGIONS</name>
19         <persName type="director">Béatrice Joyeux-Prunel</persName>
20         <orgName>UNIGE</orgName>
21       <address>
22         <addrLine>Rue des Battoirs 7</addrLine>
23         <postCode>1205</postCode>
24         <settlement>Genève</settlement>
25       </address>
26     </publisher>
27     <date when="2020"/>
28     <availability><!-- Licence texte -->
29   </publicationStmt>
30 </fileDesc>
31 </teiHeader>
32 <!-- value of attribute "when" is invalid; must be a year, must be an ISO month (of the form --MM), must be an ISO day of the month (of the form --DD), must be an ISO
33 -->

```

4. Correct the document until it is green.

- First, you have to enter the date of use in the <application> tags. You can just fill the year.

```

TEI teiHeader revisionDesc change #comment
105   <language><!-- La valeur de l'attribut "ident" correspondra à la colonne "Langue_source" (Tableau_prototype_catexp_periodiques) -->
106     <language ident="fr"/>
107   </language>
108 </profileDesc>
109 <encodingDesc xmlns:tei="http://www.tei-c.org/ns/1.0"
110   xmlns="http://purl.oclc.org/dsdl/schematron">
111   <samplingDecl>
112     <p>This electronic version of the catalog only reproduces the entries that
113       correspond to exhibited works. All text preceding or succeeding the list
114       of documents is not reproduced below.</p>
115   </samplingDecl>
116   <appInfo>
117     <application version="1.11" ident="Transkribus" when="2020">
118       <label>Transkribus</label>
119       <ptr target="https://transkribus.eu/Transkribus/">
120     </application>
121     <application version="0.5.6" ident="GROBID" when="2020">
122       <label>GROBID_Dictionaries - A machine learning software for structuring
123         digitized dictionaries</label>
124       <ptr target="https://github.com/MedKhem/grobid-dictionaries/">
125     </application>
126   </appInfo>
127 </encodingDesc>

```

- Then you have to correct all the mistakes in the catalogue entries.
- Finally, look over the encoding carrefully to correct the last mistakes.

5. Rename the document `exhibCat_NAME_OF_THE_CATALOGUE.xml` in the TEI folder of the corresponding catalogue and delete all the other `.xml` files.

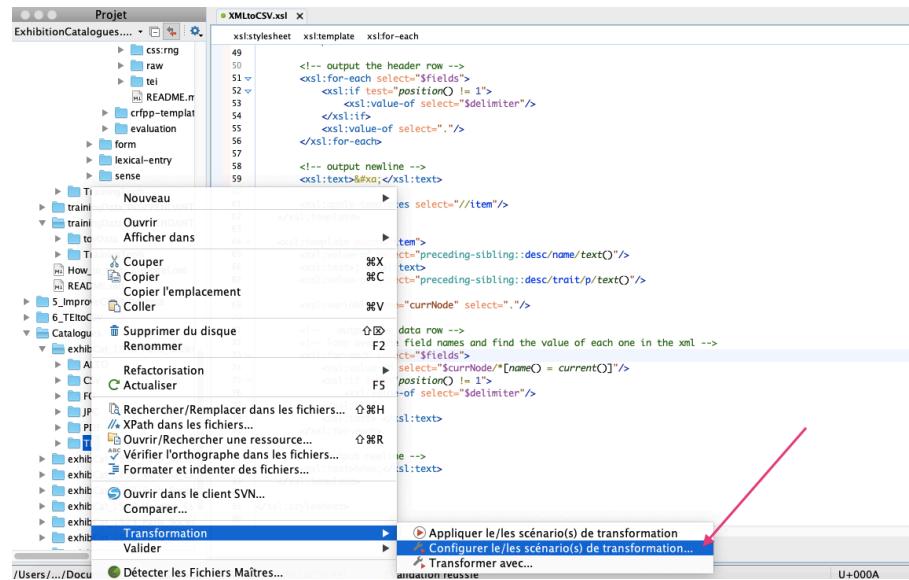
B.7 Step 6 : Transformation of the XML-TEI encoded catalogue to a CSV file

To put the data in BasArt database, we need to have them in a CSV file. To do that, we have to transform the XML-TEI version of the catalogue to a CSV file thanks to the `XMLtoCSV.xsl` xsl stylesheet. It puts the encoded data in the corresponding column of the CSV document.

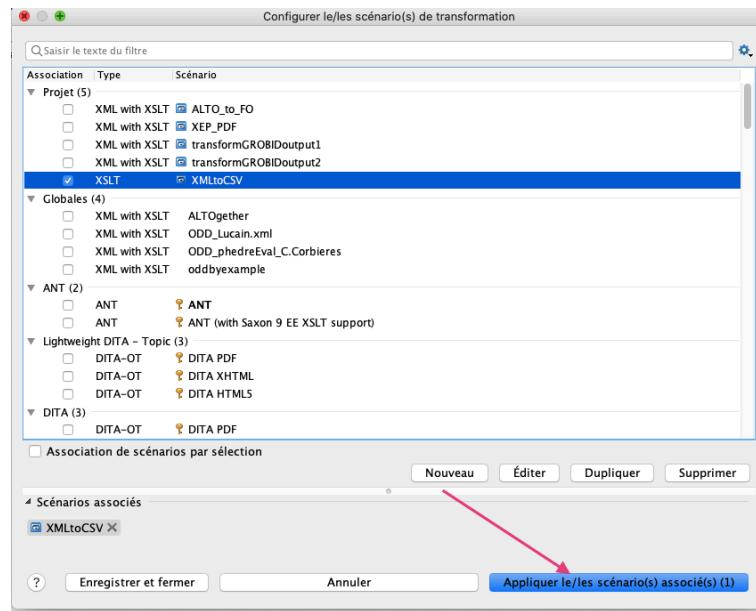
B.7.1 Transformation

Open the `ExhibitionCatalogues.xpr` project in Oxygen.

1. Go to Catalogues > `exhibCat_NAME_OF_THE_CATALOGUE` > TEI and click on Transformation > Configurer le/les scénario(s) de transformation....



2. Then select the `TEItoCSV` scenario and click on `Appliquer le/les scénario(s) associé(s)`.



3. Move the CSV file created to CSV folder.

Now you can complete the empty columns by moving the data in the corresponding columns. At this step, check for errors and correct them.

Annexe C

Guide pour entraîner des modèles sur *GROBID-dictionaries*

If you want to add training data to an existing model or train a new one, you can follow this guide. You can also refer to the official GROBID-dictionaries guidelines here¹.

This guide allow you to train PDF files.

C.1 Prepare training data

To train a model, you need to have training data for the training of the model and its evaluation. We usually use 80% data for training and 20% for evaluation.

For this tutorial, you will need `cpdf` tool. To install it, you can refer to the Step 3².

1. Choose 8 pages for the training and 2 pages for the evaluation.

2. Run in your terminal the following commands :

— Activate the `cpdf` tool.

```
cd YOUR_PATH_TO_THE_FOLDER/cpdf-binaries-master/OSX-Intel/
```

— Split your pdf.

```
./cpdf -split ~/YOUR_PATH_TO_THE_FOLDER/visual-contagions/Catalogues/  
Catalogues/exhibCat_NAME_OF_THE_CATALOGUE/PDF/exhibCat_NAME_OF_THE_  
CATALOGUE-F0.pdf -o ~/YOUR_PATH_TO_THE_FOLDER/visual-contagions/  
Catalogues/Catalogues/exhibCat_NAME_OF_THE_CATALOGUE/PDF/%%%_.pdf
```

— Rebuild a pdf with the pages you have selected. Don't forget to fill the PAGE_NUMBER of the command. To create a train document, you need to substitute `_test.pdf` for `_train.pdf` and add 6 more pages (`~/YOUR_PATH_TO_THE_FOLDER/visual-contagions/Catalogues/Catalogues/exhibCat_NAME_OF_THE_CATALOGUE/PDF/PAGE_NUMBER.pdf`).

```
./cpdf -merge -i ~/YOUR_PATH_TO_THE_FOLDER/visual-contagions/Catalogues/
```

1. <<https://github.com/MedKhem/grobid-dictionaries/wiki>>

2. <https://gitlab.unige.ch/Beatrice.Joyeux-Prunel/visual-contagions/-/blob/master/Catalogues/3_ALTOtoPDF>

```
Catalogues/exhibCat_NAME_OF_THE_CATALOGUE/PDF/PAGE_NUMBER.pdf ~/YOUR_PATH_TO_THE_FOLDER/visual-contagions/Catalogues/Catalogues/exhibCat_NAME_OF_THE_CATALOGUE/PDF/PAGE_NUMBER.pdf -o ~/YOUR_PATH_TO_THE_FOLDER/visual-contagions/Catalogues/4_GROBID/exhibCat_NAME_OF_YOUR_CATALOGUE_test.pdf
```

The files are directly created in the 4_GROBID folder.

3. Move your files to the corresponding folder.

- If you want to add training data to an existing model, you can move the files exhibCat_NAME_OF_YOUR_CATALOGUE_train.pdf and exhibCat_NAME_OF_YOUR_CATALOGUE_test.pdf in the TrainingTools/TestTrainingPDF/PDF_testFiles folder and in the toyData/dataset/dictionary-segmentation/corpus/pdf folder.
- If you want to train a new model, create a new folder named trainingData_MODEL_NAME_PDF in 4_GROBID folder. Then, move your files in the TrainingTools/TestTrainingPDF/PDF_testFiles folder and in the toyData/dataset/dictionary-segmentation/corpus/pdf folder.

C.2 Train a model with PDF files

C.2.1 Level 1 : Dictionary segmentation

1. Synchronise the right train set.

```
docker run -v PATH_TO_YOUR_FOLDER/visual-contagions/Catalogues/4_GROBID/trainingData_NAME_OF_THE_DATASET/toyData:/grobid/grobid-dictionaries/resources -p 8080:8080 -it medkhem/grobid-dictionaries-stable bash
```

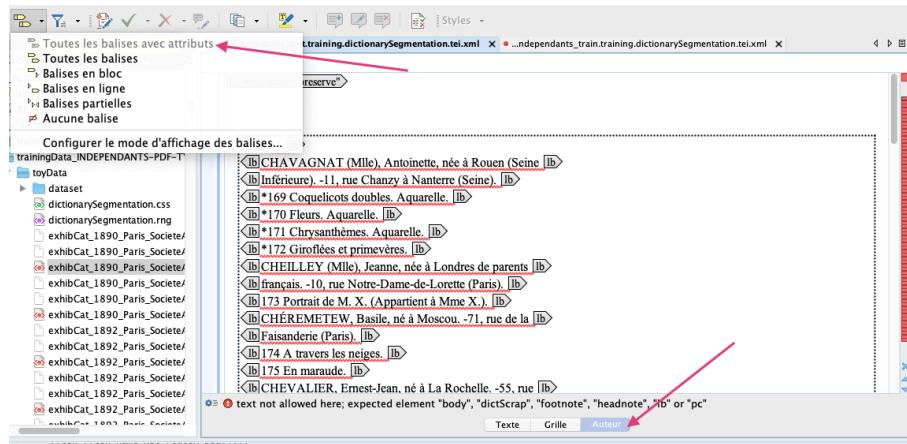
2. Create the training data.

```
java -jar /grobid/grobid-dictionaries/target/grobid-dictionaries-0.5.4-SNAPSHOT-one-jar.jar -dIn resources/dataset/dictionary-segmentation/corpus/pdf/ -dOut resources -exe createTrainingDictionarySegmentation
```

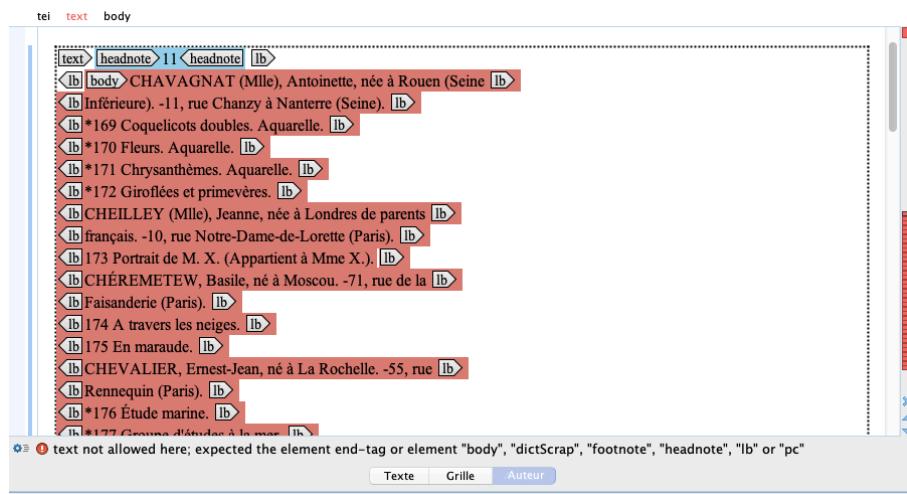
This command will create 5 different files (.css, .rng, .xml, .rawtxt and .training.dictionarySegmentation) in the toyData folder.

3. Annotate the XML-TEI file(s).

- Open the .xml file in Oxygen.
- Click on the Auteur tab and select Toutes les balises avec attributs on the upper left-hand corner.



- Then annotate the file with the `<headnote>`, `<body>` and `<footnote>` tags. Select the text you want to annotate, type cmd + E and choose the corresponding tag.



4. Move the training files in the corresponding folder of the `dataset/dictionary-segmentation/corpus` folder.

- Move the `.xml` file in the `tei` folder.
- Move the `.css` and `.rng` files in the `css/rng` folder.
- Move the `.rawtxt` and `.training.dictionarySegmentation` files in the `raw` folder.

5. Move the evaluation files in the corresponding folder of the `dataset/dictionary-segmentation/evaluation` folder.

- Move the `.xml` file in the `tei` folder.
- Move the `.css` and `.rng` files in the `css/rng` folder.

6. Run the learning process

```
mvn generate-resources -P train_dictionary_segmentation -e
```

C.2.2 Level 2 : Dictionary body segmentation

1. Create the training data.

```
java -jar /grobid/grobid-dictionaries/target/grobid-dictionaries-0.5.4
```

```
-SNAPSHOT.one-jar.jar -dIn resources/dataset/dictionary-segmentation/
corpus/pdf/ -dOut resources -exe createTrainingDictionaryBodySegmentation
```

This command will create 5 different files (.css, .rng, .xml, .rawtxt and .dictionaryBodySegmentation) in the toyData folder.

2. Annotate the XML-TEI file(s).

- Open the .xml file in Oxygen.
- Click on the Auteur tab and select Toutes les balises avec attributs on the upper left-hand corner.
- Then annotate the file with the <entry> tag. Select the text you want to annotate, type cmd + E and choose the corresponding tag.



3. Move the training files in the corresponding folder of the dataset/dictionary-body-segmentation/corpus folder.

- Move the .xml file in the tei folder.
- Move the .css and .rng files in the css/rng folder.
- Move the .rawtxt and .dictionaryBodySegmentation files in the raw folder.

4. Move the evaluation files in the corresponding folder of the dataset/dictionary-body-segmentation/evaluation folder.

- Move the .xml file in the tei folder.
- Move the .css and .rng files in the css/rng folder.

5. Run the learning process

```
mvn generate-resources -P train_dictionary_body_segmentation -e
```

C.2.3 Level 3 : Lexical entry

1. Create the training data.

```
java -jar /grobid/grobid-dictionaries/target/grobid-dictionaries-0.5.4
-SNAPSHOT.one-jar.jar -dIn resources/dataset/dictionary-segmentation
/corpus/pdf/ -dOut resources -exe createTrainingLexicalEntry
```

This command will create 5 different files (.css, .rng, .xml, .rawtxt and .lexicalEntry) in the toyData folder.

2. Annotate the XML-TEI file(s).

- Open the .xml file in Oxygen.
- Click on the Auteur tab and select Toutes les balises avec attributs on the upper left-hand corner.
- Then annotate the file with the <lemma> and <sense> tags. Select the text you want to annotate, type cmd + E and choose the corresponding tag.



3. Move the training files in the corresponding folder of the dataset/lexical-entry/corpus folder.

- Move the .xml file in the tei folder.
- Move the .css and .rng files in the css/rng folder.
- Move the .rawtxt and .lexicalEntry files in the raw folder.

4. Move the evaluation files in the corresponding folder of the dataset/lexical-entry/evaluation folder.

- Move the .xml file in the tei folder.
- Move the .css and .rng files in the css/rng folder.

5. Run the learning process

```
mvn generate-resources -P train_lexicalEntries -e
```

C.2.4 Level 4 : Form

1. Create the training data.

```
java -jar /grobid/grobid-dictionaries/target/grobid-dictionaries-0.5.4
-SNAPSHOT.one-jar.jar -dIn resources/dataset/dictionary-segmentation
/corpus/pdf/ -dOut resources -exe createTrainingForm
```

This command will create 5 different files (.css, .rng, .xml, .rawtxt and .training.form) in the toyData folder.

2. Annotate the XML-TEI file(s).

- Open the .xml file in Oxygen.
- Click on the Auteur tab and select Toutes les balises avec attributs on the upper left-hand corner.
- Then annotate the file with the <name> and <desc> tags. Select the text you want to annotate, type cmd + E and choose the corresponding tag.



3. Move the training files in the corresponding folder of the dataset/form/corpus folder.

- Move the .xml file in the tei folder.
- Move the .css and .rng files in the css/rng folder.
- Move the .rawtxt and .training.form files in the raw folder.

4. Move the evaluation files in the corresponding folder of the dataset/form/evaluation folder.

- Move the .xml file in the tei folder.
- Move the .css and .rng files in the css/rng folder.

5. Run the learning process

```
mvn generate-resources -P train_form -e
```

C.2.5 Level 5 : Sense

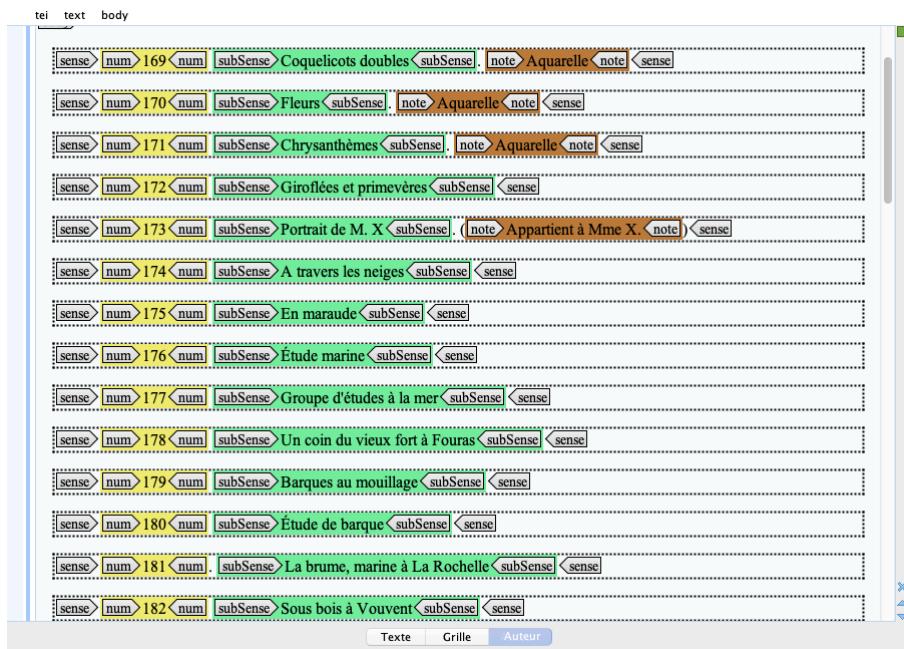
1. Create the training data.

```
java -jar /grobid/grobid-dictionaries/target/grobid-dictionaries-0.5.4
-SNAPSHOT.one-jar.jar -dIn resources/dataset/dictionary-segmentation
/corpus/pdf/ -dOut resources -exe createTrainingSense
```

This command will create 5 different files (.css, .rng, .xml, .rawtxt and .training.sense) in the toyData folder.

2. Annotate the XML-TEI file(s).

- Open the .xml file in Oxygen.
- Click on the Auteur tab and select Toutes les balises avec attributs on the upper left-hand corner.
- Then annotate the file with the <num>, <subSense> and <note> tags. Select the text you want to annotate, type cmd + E and choose the corresponding tag.



3. Move the training files in the corresponding folder of the dataset/sense/corpus folder.

- Move the .xml file in the tei folder.
- Move the .css and .rng files in the css/rng folder.
- Move the .rawtxt and .training.sense files in the raw folder.

4. Move the evaluation files in the corresponding folder of the dataset/sense/evaluation folder.

- Move the .xml file in the tei folder.
- Move the .css and .rng files in the css/rng folder.

5. Run the learning process

```
mvn generate-resources -P train_sense -e
```


Table des figures

| | | |
|-----|---|----|
| 1.1 | Capture d'écran d'un résultat de recherche sur BasArt, août 2020. | 7 |
| 2.1 | Exemple d'une entrée encodée du catalogue de la <i>Société des artistes indépendants</i> | 19 |
| 2.2 | Exemple des différents niveaux d'une entrée de catalogue d'exposition. | 21 |
| 2.3 | Exemple des différents niveaux d'une entrée de catalogue de vente de manuscrits. | 22 |
| 2.4 | Exemple d'une entrée de catalogue d'exposition encodée. | 22 |
| 2.5 | Exemple d'une entrée de catalogue de vente de manuscrits encodée. | 23 |
| 3.1 | Schéma du modèle CRF en cascade de <i>GROBID-dictionaries</i> | 29 |
| 3.2 | Capture d'écran de l'interface graphique de <i>GROBID-dictionaries</i> , août 2020. | 30 |
| 3.3 | Capture d'écran d'un fichier ALTO tiré d'un catalogue de vente de manuscrits, août 2020. | 33 |
| 3.4 | Exemple d'entrée du catalogue <i>Konstruktivisten</i> , Kunsthalle Basel, 1937. . | 35 |
| 4.1 | Capture d'écran du jeu de données du modèle TYPO_18_04_2020 entraîné dans Transkribus, juin 2020. | 38 |
| 4.2 | Capture d'écran de la transcription avec les balises d'un catalogue d'exposition dans Transkribus, août 2020. | 39 |
| 4.3 | Capture d'écran de l'évaluation du modèle HTR, septembre 2020. | 40 |
| 5.1 | Règle XSL qui applique le style typographique approprié à chaque mot. . . | 51 |
| 5.2 | Exemple d'entrée du Catalogue de la 37e exposition 1926 | 51 |
| 5.3 | Exemple d'entrée du Catalogue de la 37e exposition 1926, version PDF . . | 52 |
| 6.1 | Capture d'écran du mode « auteur » d'Oxygen, septembre 2020. | 54 |
| 6.2 | Capture d'écran d'un résultat d'entraînement du niveau <i>lexical entry</i> dans <i>GROBID-dictionaries</i> , septembre 2020. | 55 |
| 7.1 | Schéma récapitulant les 6 étapes du <i>workflow</i> | 64 |
| 7.2 | Capture d'écran du serveur local de <i>GROBID-dictionaries</i> , septembre 2020. | 67 |

| | | |
|------|---|----|
| 9.1 | Proportion des artistes femmes au salon de la Société des artistes indépendants de 1890 | 78 |
| 9.2 | Proportion des artistes femmes au salon de la Société des artistes indépendants de 1892 | 79 |
| 9.3 | Proportion des artistes femmes au salon de la Société des artistes indépendants de 1923 | 79 |
| A.1 | Exemple d'entrée du <i>Catalogue de l'exposition des oeuvres de Claude Monet</i> | 91 |
| A.2 | Exemple d'entrée du catalogue <i>Exposition des oeuvres de Gustave Courbet</i> | 91 |
| A.3 | Exemple d'entrée du catalogue <i>II bienal de São Paulo</i> | 92 |
| A.4 | Exemple d'entrée du catalogue <i>VI. Esposizione Internazionale d'Arte Della Città Di Venezia</i> | 92 |
| A.5 | Exemple d'entrée du catalogue de l' <i>Exposition chinoise d'art ancien et moderne</i> | 92 |
| A.6 | Exemple d'entrée du <i>Catalogue of the inaugural exhibition</i> | 93 |
| A.7 | Exemple d'entrée du <i>Catalogue de l'exposition des oeuvres d'art d'artistes vivants</i> | 93 |
| A.8 | Exemple d'entrée du <i>Catalogue des peintures, miniatures, aquarelles, dessins, sculptures et lithographies exposés à Nancy</i> | 93 |
| A.9 | Exemple d'entrée du <i>Catalogue de la 22e exposition 1906</i> | 94 |
| A.10 | Exemple d'entrée du catalogue <i>The exhibition of the Royal Academy of Arts</i> | 94 |
| A.11 | Exemple d'entrée du <i>Catalogue illustré de la section japonaise à l'exposition internationale des arts décoratifs et industriels modernes</i> | 94 |

Liste des tableaux

| | | |
|-----|---|----|
| 4.1 | Résultats de l'évaluation du modèle HTR (Catalogues d'exposition) | 42 |
| 4.2 | Résultats de l'évaluation du modèle HTR (Catalogues de vente de manuscrits) | 43 |
| 6.1 | Résultats du modèle <code>trainingData_INDEPENDANTS-PDF-TYPO</code> | 56 |
| 6.2 | Résultats du modèle <code>trainingData_INDEPENDANTS-PDF-ST</code> | 57 |
| 6.3 | Résultats du modèle <code>trainingData_INDEPENDANTS-ALTO</code> | 58 |
| 6.4 | Résultats du modèle <code>trainingData_LAC_LAV_typo</code> | 59 |

Table des matières

| | |
|---|-----------|
| Résumé | iii |
| Remerciements | v |
| Bibliographie | vii |
| Liste des acronymes | xv |
| Introduction | xvii |
| I Encoder automatiquement des catalogues : un enjeu inter-institutionnel | 1 |
| 1 Équipes et projets | 3 |
| 1.1 Le projet Artl@s | 3 |
| 1.1.1 Un projet de plus de dix ans | 3 |
| 1.1.2 La base de données BasArt | 5 |
| 1.2 Autres équipes | 8 |
| 1.2.1 L'équipe-projet ALMAnaCH | 8 |
| 1.2.2 Le projet e-ditiones | 9 |
| 2 Les sources : typologie et encodage | 13 |
| 2.1 Typologie des catalogues d'exposition | 14 |
| 2.1.1 Les expositions monographiques | 14 |
| 2.1.2 Les expositions de groupe | 15 |
| 2.2 Modélisation de l'encodage en XML-TEI | 17 |
| 2.2.1 Encoder des catalogues d'exposition | 17 |
| 2.2.2 Un modèle général pour encoder les catalogues | 20 |
| II De l'OCR à <i>GROBID-dictionaries</i> : le défi des fichiers | |

| | |
|---|-----------|
| ALTO | 25 |
| 3 Intégration des fichiers ALTO | 27 |
| 3.1 <i>GROBID-dictionaries</i> : fonctionnement et enjeux | 27 |
| 3.1.1 Fonctionnement général | 27 |
| 3.1.2 Intérêts et enjeux | 31 |
| 3.2 Intérêt du format ALTO | 31 |
| 3.2.1 Le format ALTO | 31 |
| 3.2.2 Intérêt de l'ALTO : l'exemple du <i>Konstruktivisten</i> | 34 |
| 4 Production de fichiers ALTO en sortie de l'OCR | 37 |
| 4.1 Conservation de l'information typographique dans l'OCR | 37 |
| 4.1.1 Entraînement d'un modèle HTR | 37 |
| 4.1.2 Évaluation du modèle | 39 |
| 4.2 Correction de l'OCR et transformation des fichiers ALTO | 44 |
| 4.2.1 Corriger l'OCR | 44 |
| 4.2.2 Insérer l'information typographique dans les fichiers ALTO | 45 |
| 5 Retard du projet et recherche de solutions | 47 |
| 5.1 Retard et difficultés du développeur de <i>GROBID-dictionaries</i> | 47 |
| 5.2 Retour vers le PDF | 49 |
| 5.2.1 Conserver l'information typographiques dans un PDF : solutions envisagées | 49 |
| 5.2.2 Le choix d'XSL-FO | 50 |
| 6 Entraînement et évaluation de modèles | 53 |
| 6.1 Entrainer un modèle dans <i>GROBID-dictionaries</i> | 53 |
| 6.2 Les modèles entraînés | 55 |
| 6.2.1 Modèles entraînés à partir des PDF | 55 |
| 6.2.2 Modèle entraîné à partir des ALTO | 57 |
| 6.2.3 Comparaison des modèles : quelle efficacité? | 59 |
| III Pérenniser le projet : proposition d'un <i>workflow</i> réutilisable | 61 |
| 7 Les étapes du <i>workflow</i> | 63 |
| 7.1 Importer les catalogues | 63 |
| 7.2 OCR et création de fichiers ALTO | 65 |
| 7.3 De l'ALTO au PDF en passant par XSL-FO | 66 |
| 7.4 Encoder automatiquement des catalogues | 67 |

| | |
|---|------------|
| TABLE DES MATIÈRES | 143 |
| 7.5 Améliorer l'encodage | 68 |
| 7.6 Du format XML-TEI au CSV | 69 |
| 8 Un <i>workflow</i> « <i>user friendly</i> » ? Choix et limites | 71 |
| 8.1 Forme et utilisateur·rice·s du <i>workflow</i> | 71 |
| 8.1.1 Qui sont les utilisateur·rice·s du <i>workflow</i> ? | 71 |
| 8.1.2 Forme et applications sélectionnées | 72 |
| 8.2 Limites de l'accessibilité du <i>workflow</i> | 74 |
| 9 Utilisations, perspectives et améliorations | 77 |
| 9.1 Exemple d'utilisations du <i>workflow</i> et premières perspectives | 77 |
| 9.1.1 Exemple d'utilisation | 77 |
| 9.1.2 Premières perspectives | 80 |
| 9.2 Open source et améliorations | 81 |
| 9.2.1 Le choix de l' <i>open source</i> | 81 |
| 9.2.2 Quelles améliorations ? | 82 |
| Conclusion | 85 |
| Annexes | 91 |
| A Sources : entrées de catalogues | 91 |
| A.1 Catalogues d'exposition monographique | 91 |
| A.2 Catalogues de biennales | 92 |
| A.3 Catalogues de Salons et de musées | 92 |
| B Workflow | 95 |
| B.1 Step 0 | 97 |
| B.1.1 Requirements | 97 |
| B.1.2 Import of the CSV files into Oxygen XML Editor | 98 |
| B.1.3 Generation of TEI headers with XSLT | 100 |
| B.2 Step 1 | 102 |
| B.2.1 Digitised Catalogues | 102 |
| B.2.2 Non digitised Catalogues | 104 |
| B.3 Step 2 | 104 |
| B.3.1 OCR process | 104 |
| B.3.2 Tag verification and correction | 110 |
| B.3.3 Transformation of the ALTO-XML files | 112 |
| B.3.4 Credits | 113 |
| B.4 Step 3 | 113 |

| | | |
|---------------------------|--|------------|
| B.4.1 | Prerequisites | 113 |
| B.4.2 | Transformation | 114 |
| B.4.3 | Thanks | 119 |
| B.5 | Step 4 | 119 |
| B.5.1 | Choose the right train set | 119 |
| B.5.2 | Encode a catalogue | 121 |
| B.5.3 | Create and train a new model | 122 |
| B.5.4 | Credits and licence | 122 |
| B.6 | Step 5 | 123 |
| B.6.1 | Transformation 1 | 123 |
| B.6.2 | Transformation 2 | 124 |
| B.7 | Step 6 | 126 |
| B.7.1 | Transformation | 126 |
| C | Guide | 129 |
| C.1 | Prepare training data | 129 |
| C.2 | Train a model with PDF files | 130 |
| C.2.1 | Level 1 : Dictionary segmentation | 130 |
| C.2.2 | Level 2 : Dictionary body segmentation | 131 |
| C.2.3 | Level 3 : Lexical entry | 132 |
| C.2.4 | Level 4 : Form | 134 |
| C.2.5 | Level 5 : Sense | 135 |
| Table des figures | | 137 |
| Liste des tableaux | | 139 |
| Table des matières | | 141 |