

ÉCOLE NATIONALE DES CHARTES
UNIVERSITÉ PARIS, SCIENCES & LETTRES

Esteban Sánchez Oeconomó

licencié ès lettres

diplômé de master

Du catalogue papier au tableau numérique.

**Un pipeline semi-automatique pour former
des étudiants en sciences humaines aux
humanités numériques**

Mémoire pour le diplôme de master

« Technologies numériques appliquées à l'histoire »

2022

Résumé

Le présent mémoire établi un premier bilan pour un stage effectué pendant l'été 2022 à l'Université de Genève, sous la direction de Béatrice Joyeux-Prunel et de Simon Gabay. Financée par l'École normale supérieure et par le centre IMAGO, cette mission s'est déroulée au sein du projet Artl@s, qui vise à cartographier les circulations artistiques mondiales des XIXe et XXe siècles.

Ce travail présente une chaîne de traitement semi-automatique pour extraire des données contenues dans des catalogues d'expositions d'art du XIXe et XXe siècles. Il s'intéresse à ses enjeux scientifiques, pédagogiques et plus largement techniques. Il propose de positionner la mission dans un processus dilaté sur trois stages effectués depuis 2019, en prenant soin de comprendre les continuités et les ruptures qui ont mené à définir une direction résolument tournée vers l'accessibilité de l'outil.

Le pipeline développé vise à produire efficacement des données pour la recherche, mais sa vocation fondamentale est pédagogique : l'outil a été conçu pour introduire des étudiants en sciences humaines aux principaux outils des humanités numériques. Ce mémoire suggère que le livrable développé mène, pour la première fois, à remplir un objectif défini depuis le début des stages : permettre aux collaborateurs du projet de produire des données satisfaisantes à partir d'un programme simple et solide qui leur transmette les notions techniques fondamentales des humanités numériques.

Mots-clés : Artl@s ; catalogues ; histoire de l'art ; OCR ; Python ; XML-ALTO ; XML-TEI ; CSV

Informations bibliographiques : Esteban Sánchez Oeconomio, *Du catalogue papier au tableau numérique. Un pipeline semi-automatique pour former des étudiants en sciences humaines aux humanités numériques*, mémoire de master « Technologies numériques appliquées à l'histoire », dir. Béatrice Joyeux-Prunel et Simon Gabay, École nationale des chartes, 2022.

Remerciements

”Tu finis ton master, hein ?”

JE dédie mon travail à ma mère, à qui j'ai promis que je terminerais cette formation.

Ce travail, et de manière plus large l'ensemble de mon parcours de M2, n'auraient pas été menés à terme sans l'aide spontanée de l'équipe pédagogique et administrative du master TNAH et de mes responsables de stage. Leur compréhension m'a permis de tenir un engagement que j'ai cru à certains moments inatteignable.

Je voudrais donc remercier Mr. Thibault Clérice, qui n'a pas hésité à prendre des mesures adaptées à une situation particulière. Je souhaite aussi remercier certains enseignants pour la gentillesse avec laquelle ils ont facilité la mise en place de ces mesures : Mme Ariane Pinche, Mme Ségolène Albouy, Mr. Edward Gray.

Je suis très reconnaissant envers Mme Béatrice Joyeux-Prunel et Mr. Simon Gabay, dont la patience a été tout particulièrement aidante pendant mon stage, lors de discussions stimulantes mais aussi lors de l'organisation logistique de ma mission.

Pour terminer, je remercie Laura, profondément, de m'avoir accompagné pendant cette aventure académique ponctuée de travail et de fortes émotions.

Bibliographie

Catalogues traités (corpus Artl@s)

ASSOCIATION FRANÇAISE D'ACTION ARTISTIQUE, *Deuxième Biennale de Paris : Manifestation Biennale et Internationale Des Jeunes Artistes Du 29 Septembre Au 5 Novembre 1961, Musée d'art Moderne de La Ville de Paris Org. Sous Les Auspices de l'Association Française d'action Artistique, Avec La Participation de La Radio-diffusion-télévision Française*, 1961.

- *Quatrième Biennale de Paris : Manifestation Biennale et Internationale Des Jeunes Artistes : Du 29 Septembre Au 3 Novembre 1965 : Musée d'Art Moderne de La Ville de Paris*, 1965.
- *Sixième Biennale de Paris : Manifestation Biennale et Internationale Des Jeunes Artistes : Du 2 Octobre Au 2 Novembre 1969 : Musée d'Art Moderne de La Ville de Paris*, 1969.

Bienal Do Museu de Arte Moderna de São Paulo. Octubro a Dezembro, São Paulo, 1951,
URL : <https://issuu.com/bienal/docs/name3fe634>.

Jules-Antoine Castagnary (éd.), *Exposition Des Oeuvres de Gustave Courbet à l'école Des Beaux-Arts (Mai 1882)*, Paris, 1882, URL : <https://gallica.bnf.fr/ark:/12148/bpt6k6574415d?rk=21459;2>.

Catalogue de La Dix-Huitième Exposition Annuelle Du Musée de Rouen, Rouen, 1860,
URL : <https://gallica.bnf.fr/ark:/12148/bpt6k1181800p.r=Catalogue%20de%20La%20Dix-Huiti%C3%A8me%20Exposition%20Annuelle%20Du%20Mus%C3%A9e%20Rouen?rk=21459;2>.

Catalogue de La Quinzième Exposition Annuelle Du Musée de Rouen, Rouen, 1853, URL : <https://gallica.bnf.fr/ark:/12148/bpt6k1181799g.image>.

Catalogue de La Seizième Exposition Annuelle Du Musée de Rouen, Rouen, 1856, URL : <https://gallica.bnf.fr/ark:/12148/bpt6k1181800p>.

Catalogue de La Trente-Deuxième Exposition Annuelle Du Musée de Rouen, Rouen, 1891,
URL : <https://gallica.bnf.fr/ark:/12148/bpt6k1181805r?rk=21459;2>.

Catalogue de La Trente-Quatrième Exposition Annuelle Du Musée de Rouen, Rouen, 1895,
URL : <https://gallica.bnf.fr/ark:/12148/bpt6k11806295?rk=21459;2>.

- Catalogue de La Trente-Unième Exposition Annuelle Du Musée de Rouen*, Rouen, 1888,
URL : <https://gallica.bnf.fr/ark:/12148/bpt6k1181804b?rk=21459;2>.
- Catalogue de La Vingt-Deuxième Exposition Annuelle Du Musée de Rouen*, Rouen, 1869,
URL : <https://gallica.bnf.fr/ark:/12148/bpt6k1181045h?rk=21459;2>.
- Catalogue de La Vingt-Sixième Exposition Annuelle Du Musée de Rouen*, Rouen, 1878,
URL : <https://gallica.bnf.fr/ark:/12148/bpt6k11818013?rk=21459;2>.
- Catalogue de La Vingt-Troisième Exposition Annuelle Du Musée de Rouen*, Rouen, 1872,
URL : <https://gallica.bnf.fr/ark:/12148/bpt6k1181046x?rk=21459;2>.
- Catalogue Des Ouvrages de Peinture, Sculpture, Gravure, Lithographie et Architecture : Refusés Par Le Jury de 1863 et Exposés, Par Décision de S.M. l'Empereur Au Salon Annexe, Palais Des Champs-Elysées, Le 15 Mai 1863*, Paris, 1863, URL : <https://gallica.bnf.fr/ark:/12148/bpt6k114147v>.
- Catalogue Du Salon de La Rose † Croix : Geste Esthétique*, Paris, 1893, URL : <https://gallica.bnf.fr/ark:/12148/bpt6k5470406r.r=catalogue+salon.langFR#>.
- Dodicesima Esposizione Internazionale d'arte Della Città Di Venezia. Catalogo Illustrato*, Venezia, 1920, URL : <https://catalog.hathitrust.org/Record/000642649>.
- MUSÉE IMPÉRIAL DU LUXEMBOURG, *Notice de peintures, sculptures et dessins de l'école moderne de France exposées dans les galeries du Musée impérial du Luxembourg*, 1867, URL : <https://gallica.bnf.fr/ark:/12148/bpt6k64243519>.
- MUSÉE ROYAL DU LUXEMBOURG, *Explication des ouvrages de peinture et sculpture, de l'école moderne de France, exposés le 24 avril 1818, dans le musée royal du Luxembourg, destiné aux artistes vivants*, 1820, URL : <https://gallica.bnf.fr/ark:/12148/bpt6k6491364x>.
- Museu de arte moderna et Biennale internationale de São Paulo (éd.), *II Bienal de São Paulo, Museu de Arte Moderna*, São Paulo, Brésil, 1953, URL : <https://issuu.com/bienal/docs/name24c514>.
- PHOTO-CLUB DE PARIS, *Catalogue de La Première Exposition d'art Photographie*, Paris, 1894, URL : <https://libmma.contentdm.oclc.org/digital/collection/p15324coll19/id/2640/rec/5>.
- *Catalogue de La Troisième Exposition d'art Photographique*, Paris, 1896, URL : <https://libmma.contentdm.oclc.org/digital/collection/p15324coll19/id/3090/rec/6>.
- *Catalogue Des Œuvres Exposées Au Cinquième Salon International de Photographie*, Paris, 1898, URL : <https://libmma.contentdm.oclc.org/digital/collection/p15324coll19/id/3663/rec/3>.
- *Catalogue Des Œuvres Exposées Au Septième Salon International de Photographie*, Paris, 1902, URL : <https://libmma.contentdm.oclc.org/digital/collection/p15324coll19/id/307/rec/1>.

- *Catalogue Des Œuvres Exposées Au Neuvième Salon International de Photographie*, Paris, 1904, URL : <https://libmma.contentdm.oclc.org/digital/collection/p15324coll19/id/5249/rec/9>.
- *Catalogue Des Œuvres Exposées Au Onzième Salon International de Photographie*, Paris, 1906, URL : <https://libmma.contentdm.oclc.org/digital/collection/p15324coll19/id/7055/rec/10>.

Prima Esposizione Internazionale d'Arte Della Città Di Venezia. Catalogo Illustrato, Venezia, 1895, URL : [https://babel.hathitrust.org/cgi/pt?id=uc1.\\$b203824&view=1up&seq=7](https://babel.hathitrust.org/cgi/pt?id=uc1.$b203824&view=1up&seq=7).

SALON D'AUTOMNE, SOCIÉTÉ DES ARTISTES DÉCORATEURS et SALON DES TUILERIES, *Catalogue Des Ouvrages de Peinture, Sculpture, Dessin... Exposés Au Palais de Chaillot... Du 3 Avril Au 25 Avril 1940*, Paris, 1940, URL : <https://gallica.bnf.fr/ark:/12148/bpt6k1179734b/f1.image>.

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]*, Paris, 1892, URL : <https://gallica.bnf.fr/ark:/12148/bpt6k1269951h>.

- *Société Des Artistes Indépendants. 22, Catalogue de La 22e Exposition 1906 : Grandes Serres de La Ville de Paris... Du 20 Mars Au 30 Avril... Dir. L'Emancipatrice*, Paris, 1906, URL : <https://gallica.bnf.fr/ark:/12148/bpt6k9687742d>.
- *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*, Paris, 1913, URL : <https://gallica.bnf.fr/ark:/12148/bpt6k8892045>.
- *Société Des Artistes Indépendants. 34, Catalogue de La 34e Exposition 1923 : Grand Palais Des Champs-Elysées, Du 10 Février Au 11 Mars*, Paris, 1923.
- *Société Des Artistes Indépendants. 37, Catalogue de La 37e Exposition 1926... Du 20 Mars Au 2 Mai*, Paris, 1926, URL : <https://gallica.bnf.fr/ark:/12148/bpt6k9656486d> (visité le 06/09/2020).
- *Société Des Artistes Indépendants. 46, Catalogue de La 46e Exposition 1935 : Au Grand Palais Des Champs-Elysées Du 18 Janvier Au 3 Mars Inclus*, Paris, 1935, URL : <https://gallica.bnf.fr/ark:/12148/bpt6k1423039g.r=salon%20des%20ind%C3%A9pendants?rk=665239;2>.

Sesta Esposizione Internazionale d'Arte Della Città Di Venezia, 1905 : Catalogo Illustrato, Venezia, 1905, URL : https://primo.getty.edu/primo-explore/fulldisplay/GETTY_ALMA21137033710001551/GRI.

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, URL : <https://gallica.bnf.fr/ark:/12148/bpt6k9105588p?rk=21459;2>.

- SOCIÉTÉ DES ARTISTES FRANÇAIS, *Catalogue Illustré Du Salon de 1887*, dir. F.-G. Dumas, Paris, 1887, URL : <https://gallica.bnf.fr/ark:/12148/bpt6k2024955.r=Catalogue%20Illustr%C3%A9%20Du%20Salon%20de%201887?rk=21459;2>.
- SOCIÉTÉ LORRAINE DES AMIS DES ARTS, *Catalogue Des Peintures, Miniatures, Aquarelles, Dessins, Sculptures et Lithographies Exposés à Nancy... Par Les Artistes Lorrains*, Nancy, 1843, URL : <https://gallica.bnf.fr/ark:/12148/bpt6k62251805.texteImage>.
- *Catalogue Des Peintures, Miniatures, Aquarelles, Dessins, Sculptures et Lithographies Exposés à Nancy... Par Les Artistes Lorrains*, Nancy, 1847, URL : <https://gallica.bnf.fr/ark:/12148/bpt6k62275056?rk=21459;2>.
 - *Catalogue Des Peintures, Miniatures, Aquarelles, Dessins, Sculptures et Lithographies Exposés à Nancy... Par Les Artistes Lorrains*, Nancy, 1849, URL : <https://gallica.bnf.fr/ark:/12148/bpt6k62276311?rk=21459;2>.
 - *Catalogue Des Peintures, Miniatures, Aquarelles, Dessins, Sculptures et Lithographies Exposés à Nancy... Par Les Artistes Lorrains*, Nancy, 1892, URL : <https://gallica.bnf.fr/ark:/12148/bpt6k6229011t?rk=21459;2>.
- XII Bienal de São Paulo*, 1973, URL : <https://issuu.com/bienal/docs/namée1bdf4/440>.

Catalogues non traités (corpus Artl@s)

- Léonce Bénédite (éd.), *Exposition Charles Cottet : Catalogue*, 1911.
- Catalogue de l'exposition Des Oeuvres 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.
- MONTROSS GALLERY et MATISSE (Henri), *Henri Matisse Exhibition, January 20th to February 27, 1915, Catalogue*, New York, 1915.
- 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*, dir. Harold B. Lee, New York, 1912.

Histoire globale, histoire de l'art et histoire des catalogues

- ALBANO (Caterina), *Exhibition*, dir. Oxford University, Oxford et New York, URL : <https://ualresearchonline.arts.ac.uk/id/eprint/8058/>.

- 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 de Geneve, Editions des Cendres, 2015 (Bibliothèques).
- BAYLY (Christopher Alan), BECKERT (Sven), CONNELLY (Matthew), HOFMEYR (Isabel), KOZOL (Wendy) et SEED (Patricia), « AHR Conversation : On Transnational History », *The American Historical Review*, 111–5 (2006), p. 1441-1464, DOI : 10.1086/ahr.111.5.1441.
- BERTRAND DORLÉAC (Laurence), *Le Commerce de l'art : De La Renaissance à Nos Jours*, Editions La Manufacture, Besançon, 1992.
- BESSE (Jean Marc), « Approches Spatiales Dans l'histoire Des Sciences et Des Arts », *L'Espace géographique*, 39 (2010), p. 211-224.
- BON (François), *Après Le Livre*, Paris, 2011.
- BON (Laurent Le) et HUESCA (Roland), « Propos Autour Du Catalogue d'expo.... Entretien Avec Roland Huesca », *Le Portique. Revue de philosophie et de sciences humaines*-30 (juill. 2013), DOI : 10.4000/leportique.2638.
- CHANTE (Alain), « La notion de catalogue : de l'imprimé au numérique », *Culture & Musées*, 21–1 (2013), p. 131-152, DOI : 10.3406/pumus.2013.1735.
- CHAUBET (François), *La mondialisation culturelle*, Paris, 2013, URL : http://www.cairn.info/numero.php?ID_NUMPUBLIE=PUF_CHAUB_2013_01 (visité le 18/07/2019).
- DACOSTA KAUFMANN (Thomas), *Towards a Geography of Art*, Chicago, 2004.
- DASTON (Lorraine) et GALISON (Peter), *Objectivity*, New York, 2007.
- DOSSIN (Catherine), « Towards a Spatial (Digital) Art History », *Artl@s Bulletin*, 4–1 (11 juin 2015), URL : <https://docs.lib.psu.edu/artlas/vol4/iss1/1>.
- ELKINS (James), *Is Art History Global?*, New York, 2007.
- FICKERS (Andreas), « Experimental Media Archaeology : A Plea for New Directions », *Annie van den Oever (ed.). Technē/Technology Researching Cinema and Media Technologies – Their Development, Use, and Impact* (Amsterdam : Amsterdam University Press, 2013), pp. 272-278. (), URL : https://www.academia.edu/5578016/Experimental_Media_Archaeology_A_Plea_for_New_Directions (visité le 09/09/2018).
- GALOIN (Alain), *Le Salon de la Rose-Croix*, URL : <https://histoire-image.org/fr/etudes/salon-rose-croix> (visité le 18/07/2022).
- GONCERUT (Véronique), *Les Catalogues d'exposition : Gros Plan Sur Ces Ouvrages d'art Devenus Incontournables*, avec la coll. de Maureen Marozeau, mars 2017, URL : <https://blog.mahgeneve.ch/les-catalogues-dexposition/>.

- HAUCHECORNE (Mathieu), « Les « humanités scientifiques » selon Bruno Latour », *Critique*, 786–11 (2012), p. 933-948, URL : <https://www-cairn-info.proxy.chartes-psl.eu/revue-critique-2012-11-page-933.htm> (visité le 21/11/2020).
- HOUSSAIS (Laurent) et LAGRANGE (Marion), *Marché(s) de l'art En Province 1870-1914*, Presses universitaires de Bordeaux, Bordeaux, 2010 (Les Cahiers Du Centre François Georges Pariet, 8).
- Christian Jacob (éd.), *Lieux de savoir. 2 : Les mains de l'intellect*, Paris, 2010.
- 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), « L'histoire de l'art et le quantitatif », *Histoire & mesure*, XXIII–2 (2[2008]), p. 3-34, URL : <http://journals.openedition.org/histoiremesure/3543> (visité le 25/08/2022).
- « Ce que l'approche mondiale fait à l'histoire de l'art », *Romantisme*, 163–1 (2014), p. 63-78.
- « Provincializing Paris. The Center-Periphery Narrative of Modern Art in Light of Quantitative and Transnational Approaches », *Artl@s Bulletin*, 4–1 (11 juin 2015), URL : <https://docs.lib.purdue.edu/artlas/vol4/iss1/4>.
- (éd.), *L'Art et la mesure : Histoire de l'art et méthodes quantitatives*, Paris, 2022 (Actes de la recherche à l'ENS), URL : <http://books.openedition.org/editionsulm/8557> (visité le 25/08/2022).
- JOYEUX-PRUNEL (Béatrice) et DA COSTA KAUFMANN (Thomas), « Reintroducing Circulations : Historiography and the Project of Global Art History », dans *Circulations in the Global History of Art*, dir. Catherine Dossin, Béatrice Joyeux-Prunel et Thomas DaCosta Kaufmann, 2015, p. 1-22.
- JOYEUX-PRUNEL (Béatrice) et MARCEL (Olivier), « Exhibition Catalogues in the Globalization of Art. A Source for Social and Spatial Art History », 4–2 (2015), p. 26.
- KUHN (Thomas), *The Structure of Scientific Revolutions*, Chicago, 1970.
- LATOUR (Bruno), « Ces réseaux que la raison ignore : laboratoires, bibliothèques, collections », dans *Le Pouvoir des bibliothèques. La mémoire des livres en Occident*, dir. Christian Jacob et Marc Baratin, 1996, URL : https://login.proxy.bib.uottawa.ca/login?url=http://www.numilog.com/bibliotheque/BU-Ottawa/fiche_livre.asp?idprod=51632 (visité le 05/12/2020).
- LEINMAN (Colette), « Le Catalogue d'art Contemporain », *Marges. Revue d'art contemporain*–12 (avr. 2011), p. 51-63, DOI : 10.4000/marges.408.
- *Les catalogues d'expositions surréalistes à Paris entre 1924 et 1939*, 2015, URL : <https://brill.com/view/title/31570> (visité le 07/08/2022).
- LENOIR (Timothy), « The Discipline of Nature and the Nature of Disciplines », dans *Instituting Science : The Cultural Production of Scientific Disciplines*, 2022, p. 45-74, DOI : 10.1515/9781503616059-005.

- MANTION (Jean-Rémy), « Le lieu du génie. Remarques sur la géographie de l'art », *Annales*, 38–5 (1983), p. 1084-1096, DOI : 10.3406/ahess.1983.411004.
- PARCOLLET (Remi) et SZACKA (Léa-Catherine), « Écrire l'histoire des expositions : réflexions sur la constitution d'un catalogue raisonné d'expositions », *Culture & Musées*, 22–1 (2013), p. 137-162, DOI : 10.3406/pumus.2013.1755.
- PARINET (Elisabeth), *Les Ventes de Livres et Leurs Catalogues, XVIIe-XXe Siècle*, Publications de l'Ecole nationale des Chartes, Paris, 2018 (Etudes et Rencontres).
- ROCHE (Mélanie), *En Attendant "Le Jour [...] Où Il n'y Aura plus de Catalogue à Faire" : Une Histoire Matérielle Des Catalogues de Bibliothèques (1789-1993)*, Mémoire d'études de Diplôme de Conservateur des Bibliothèques, Lyon, ENSSIB, 2014, URL : <https://www.enssib.fr/bibliotheque-numerique/documents/64118-en-attendant-le-jour-ou-il-n-y-aura-plus-de-catalogue-a-faire-une-histoire-materielle-des-catalogues-de-bibliotheque-1789-1993.pdf>.
- ROSENBERG (Pierre), « L'apport Des Expositions et de Leurs Catalogues à l'histoire de l'art », *Les Cahiers du Musée national d'art moderne*–29 (1989), p. 49-56.
- SAUNIER (Pierre-Yves), « Circulations, connexions et espaces transnationaux », *Geneses* (, 2005), p. 110-126, URL : <https://halshs.archives-ouvertes.fr/halshs-00003886>.

Projet Artl@s

- ARTLAS et KATABASE, *OCRCat : Data Pour OCR*, 2021, URL : <https://github.com/katabase/OCRcat> (visité le 01/08/2022).
- DOSSIN (Catherine), KONG (Nicole Ningning) et JOYEUX-PRUNEL (Béatrice), « Applying VGI to Collaborative Research in the Humanities : The Case of ARTL@S », *Cartography and Geographic Information Science*, 44–6 (nov. 2017), p. 521-538, DOI : 10.1080/15230406.2016.1216804.
- JOYEUX-PRUNEL (Béatrice), « ARTL@S : A Spatial and Transnational Art History Origins and Positions of a Research Program », *Artl@s Bulletin*, 1–1 (15 sept. 2012), URL : <https://docs.lib.psu.edu/artlas/vol1/iss1/1>.
- « Visual Contagions, the Art Historian, and the Digital Strategies to Work on Them », *Artl@s Bulletin*–8–3 (2019), URL : <https://docs.lib.psu.edu/artlas/vol8/iss3/8>.
- 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).

JOYEUX-PRUNEL (Béatrice), DOSSIN (Catherine) et SAINT-RAYMOND (Léa), *Artl@s*, URL : <https://artlas.huma-num.fr/fr/>.

TOPALOV (Barbara), GABAY (Simon), JOYEUX-PRUNEL (Béatrice), ROMARY (Laurent) et RONDEAU DU NOYER (Lucie), « Automating Artl@s - Extracting Data from Exhibition Catalogues », dans 2020.

Stages Artl@s

CORBIÈRES (Caroline), *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, Paris, Ecole nationale des Chartes, 2020.

JANÈS (Juliette), *Du Catalogue Papier Au Numérique : Une Chaîne de Traitement Ouverte Pour l'extraction d'informations Issues de Documents Structurés*, mémoire de master « Technologies numériques appliquées à l'histoire », dir. Thibault Clérice et Béatrice Joyeux-Prunel, Paris, École nationale des chartes, 2021, URL : https://github.com/Juliettejns/Memoire_TNAH.

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 Béatrice Joyeux-Prunel, Paris, Ecole nationale des Chartes, 2019.

SANCHEZ OECOMO (Esteban), *Du Catalogue Papier Au Tableur Numérique. Un Pipeline Semi-Automatique Pour Former Des Étudiants En Sciences Humaines Aux Humanités Numériques*, mémoire de master « Technologies numériques appliquées à l'histoire », dir. Béatrice Joyeux-Prunel et Simon Gabay, Paris, École nationale des chartes, 2022.

Humanités numériques : pédagogie, recherche et enjeux épistémologiques

ALLOUCHE (Elie), « Humanités Numériques et Formation : Quels Enjeux Communs de l'École à l'Université ? », dans *Colloque DHNord 2019 "Corpus et Archives Numériques" MESHS Lille Nord de France – 16-18 Octobre 2019*, Lille, France, 2019, URL : <https://hal.archives-ouvertes.fr/hal-02321700> (visité le 25/08/2022).
— *Humanités numériques et éducation (classe virtuelle 01/07/20)*, Éducation, numérique et recherche, URL : <https://edunumrech.hypotheses.org/1797> (visité le 25/08/2022).

- *Humanités numériques et pratiques pédagogiques : journée d'étude (02/10/20)*, Éducation, numérique et recherche, URL : <https://edunumrech.hypotheses.org/2108> (visité le 25/08/2022).
- ALRAHABI (Motasem), ROE (Glenn), BORDRY (Marguerite), KOSKAS (Camille) et GAWLEY (James), « Des étudiants en lettres face aux humanités numériques : une expérience pédagogique », *Humanités numériques*–5 (5[2022]), DOI : 10.4000/revuehn.2775.
- BARTSCHERER (Thomas) et COOVER (Roderick), *Switching Codes : Thinking through Digital Technology in the Humanities and the Arts*, Chicago, 2011, URL : <http://site.ebrary.com/id/10468508> (visité le 25/08/2022).
- BERRA (Aurélien), « Faire des humanités numériques », dans *Read/Write Book 2 : Une introduction aux humanités numériques*, dir. Pierre Mounier, Marseille, 2012 (Read/Write Book), p. 25-43, URL : <http://books.openedition.org/oep/238> (visité le 25/08/2022).
- BIBLIOTHÈQUE DE L'INHA, *Bases de Données Sur Le Marché de l'art*, URL : <http://bibliotheque.inha.fr/iguana/www.main.cls?p=74469586-3948-11e2-a8f1-ac6f86effe00&v=2ca1bb8c-9a81-11ea-a5ae-5056b21d9100> (visité le 11/07/2022).
- BOURGATTE (Michaël), « Pour un humanisme numérique en éducation », *Revue française des sciences de l'information et de la communication*–10 (10[2017]), DOI : 10.4000/rfsic.2652.
- CANOPÉ 94 (Atelier), *Enseigner Les Humanités Numériques, de Quoi Parlons-Nous ?*, Humanités Numériques et Éducation, 10 juin 2021, URL : <https://medium.com/humanit%C3%A9s-num%C3%A9riques-et-%C3%A9ducation/enseigner-les-humanit%C3%A9s-num%C3%A9riques-de-quoi-parlons-nous-2b23a214c00d> (visité le 25/08/2022).
- CÉCI (Jean-François), BOURGATTE (Mickael), FERLONI (Mikael) et TESSIER (Laurent), *Quelles Humanités Numériques Pour l'éducation ?*, 2016.
- DEBOUY (Estelle) et IDMHAND (Fatiha), « Enseigner En Humanités Numériques : Comment Enseigner l'autonomie ? », dans *La Mise En Œuvre Des Humanités Numériques Dans Les Pratiques Pédagogiques En SHS*, Maison de la Recherche de l'université Paris 8, France, 2020, URL : <https://hal.archives-ouvertes.fr/hal-03313016> (visité le 25/08/2022).
- DEDIEU (Laurence) et MARIE-FRANÇOISE (Fily), *Rendre Publics Ses Jeux de Données Scientifiques*, Montpellier, CIRAD, 2015, p. 6, DOI : 10.18167/COOPIST/0059.
- DEVAUCHELLE (Bruno), *Comment le numérique transforme les lieux de savoirs : le numérique au service du bien commun et de l'accès au savoir pour tous*, Paris, 2012 (Société de la connaissance), URL : <http://catalogue.bnf.fr/ark:/12148/cb42627540h> (visité le 06/09/2022).

- DOSSIN (Catherine), « Towards a Spatial (Digital) Art History », *Artl@s Bulletin*, 4–1 (11 juin 2015), URL : <https://docs.lib.psu.edu/artlas/vol4/iss1/1>.
- FRAU-MEIGS (Divina), « Créativité, éducation aux médias et à l'information, translittératie : vers des humanités numériques », *Quaderni. Communication, technologies, pouvoir*–98 (98[2019]), p. 87-105, DOI : 10.4000/quaderni.1482.
- GEFEN (Alexandre), « Les enjeux épistémologiques des humanités numériques », *Socio. La nouvelle revue des sciences sociales*–4 (4[2015]), p. 61-74, DOI : 10.4000/socio.1296.
- HIRSCH (Brett D), *Digital Humanities Pedagogy : Practices, Principles and Politics*, 2012.
- JACQUEMIN (Bernard), SCHÖPFEL (Joachim) et FABRE (Renaud), « Libre Accès et Données de Recherche. De l'utopie à l'idéal Réaliste », *Études de communication*–52 (2019), DOI : 10.4000/edc.8468.
- JOYEUX-PRUNEL (Béatrice), « Bases de Données et Gestion de Projets En Humanités Numériques. Les Dessous Du Projet Artl@s », *Biens Symboliques / Symbolic Goods. Revue de sciences sociales sur les arts, la culture et les idées*–2 (févr. 2018), DOI : 10.4000/bssg.242.
- KAUFMANN (Lyonel), « L'enseignement des SHS est-il soluble dans les Humanités numérique ? Et inversément. » (5 sept. 2019), URL : <https://orfeo.hepl.ch/handle/20.500.12162/3281> (visité le 25/08/2022).
- 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.
- MORANDI (Franc), « À l'école des humanités numériques », *Hermès, La Revue*, 78–2 (2017), p. 96-103, DOI : 10.3917/herm.078.0096.
- PÉLISSIER (Chrysta), « Accompagner le chercheur en SHS à l'ère des humanités numériques. Des outils pour le développement d'une activité cognitive augmentée », *Les Cahiers du numérique*, 13–3-4 (2017), p. 167-194, URL : <https://www.cairn.info/revue-les-cahiers-du-numerique-2017-3-4-page-167.htm> (visité le 25/08/2022).
- PUREN (Marie), *La Numérisation Du Patrimoine. Du Projet Gutenberg à Google Arts & Culture*, Master, oct. 2020, URL : <https://hal.archives-ouvertes.fr/hal-03152774> (visité le 23/08/2022).
- TESSIER (Laurent) et BOURGATTE (Michaël), « Introduction. Enseigner et apprendre les humanités numériques : des savoirs recomposés ? », *Humanités numériques*–5 (5[2022]), DOI : 10.4000/revuehn.2999.
- THATCAMP PARIS, *Manifeste Des Digital Humanities*, 2010, URL : <https://tcp-hypotheses.org.ezpaarse.univ-paris1.fr/318>.

- UNESCO, *Vers Une Recommandation de l'UNESCO Pour La Science Ouverte*, 2019,
URL : https://en.unesco.org/sites/default/files/open_science_brochure_fr.pdf (visité le 23/08/2022).
- URFIST MÉDITERRANÉE, *Les Principes FAIR*, URL : <https://doranum.fr/enjeux-benefices/principes-fair/> (visité le 16/07/2022).
- VANHOLSBEECK (Marc), « La Notion de Science Ouverte Dans l'Espace Européen de La Recherche », *Revue française des sciences de l'information et de la communication*–11 (2017), DOI : 10.4000/rfsic.3241.
- VANLEENE (François), LYMAN-HAGER (Mary Ann) et MERCROL (Lise), « Humanités numériques et pédagogie participative : cas d'une classe expérimentale pluridisciplinaire inversée », *Éla. Études de linguistique appliquée*, 193–1 (2019), p. 91-105, DOI : 10.3917/ela.193.0091.
- WEBER (Anne) et APAHAU, ASSOCIATION DES PROFESSEURS D'ARCHÉOLOGIEET D'HISTOIRE DE L'ART DES UNIVERSITÉS, *Numérisation Des Catalogues de Ventes d'oeuvres d'art de La Bibliothèque de l'INHA*, 2014, URL : <http://blog.apahau.org/numerisation-des-catalogues-de-ventes-doeuvres-dart-de-la-bibliotheque-de-linha/> (visité le 13/08/2022).
- WILKINSON Mark D., DUMONTIER Michel, AALBERSBERG IJsbrand Jan, APPLETON Gabrielle, AXTON Myles, BAAK Arie, BLOMBERG Niklas, BOITEN Jan-Willem, DA SILVA SANTOS Luiz Bonino, BOURNE Philip E., et al., « The FAIR Guiding Principles for Scientific Data Management and Stewardship », *Scientific Data*, 3–1 (mars 2016), DOI : 10.1038/sdata.2016.18.

Humanités numériques : outils et logiciels

- ALTO Editorial Board (éd.), *ALTO : Technical Metadata for Layout and Text Objects*, 2004, URL : <https://www.loc.gov/standards/alto/> (visité le 09/01/2022).
- ALTO : Technical Metadata for Layout and Text Objects (Standards, Library of Congress)*, URL : <https://www.loc.gov/standards/alto/> (visité le 14/07/2022).
- ARTLAS et KATABASE, *OCRCat : Data Pour OCR*, 2021, URL : <https://github.com/katabase/OCRcat> (visité le 01/08/2022).
- Basart*, URL : <https://artlas.huma-num.fr/fr/bases-en-acces-libre/>.
- BELAÏD (Abdel), RANGONI (Yves) et FALK (Ingrid), *Représentation des données en XML pour l'analyse d'images de documents*, text, URL : <http://lodel.irevues.inist.fr/cide/index.php?id=147> (visité le 11/07/2022).
- BHATT (Jwalin), HASHMI (Khurram Azeem), AFZAL (Muhammad Zeshan) et STRICKER (Didier), « A Survey of Graphical Page Object Detection with Deep Neural Networks », *Applied Sciences*, 11–12 (janv. 2021), DOI : 10.3390/app11125344.

- BURNARD (Lou), *What Is the Text Encoding Initiative ? : How to Add Intelligent Markup to Digital Resources*, Marseille, 2014 (Encyclopédie Numérique), URL : <http://books.openedition.org/oep/426>.
- *Qu'est-ce que la Text Encoding Initiative ?, Marseille, 2015.*
- CHAGUÉ (Alix), *Conditions de La Mutualisation : Les Principes FAIR et HTR-United*, mai 2022, URL : <https://hal.inria.fr/hal-03685731> (visité le 30/08/2022).
- CHAGUÉ (Alix) et CLÉRICE (Thibault), « Sharing HTR Datasets with Standardized Metadata : The HTR-United Initiative », dans *Documents Anciens et Reconnaissance Automatique Des Écritures Manuscrites*, Paris, France, 2022, URL : <https://hal.inria.fr/hal-03703989> (visité le 30/08/2022).
- CHAGUÉ (Alix), CLÉRICE (Thibault) et ROMARY (Laurent), « HTR-United : Mutualisons La Vérité de Terrain ! », dans *#dhnord2021 - Publier, Partager, Réutiliser Les Données de La Recherche : Les Data Papers et Leurs Enjeux*, Lille, France, 2021, URL : <https://hal.archives-ouvertes.fr/hal-03398740> (visité le 09/01/2022).
- 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](https://doi.org/10.1007/s10032-019-00347-8).
- extractionCatalogs*, URL : <https://github.com/IMAGO-Catalogues-Jjanes/extractionCatalogs>.
- FUCHS (Patrick) et POULAIN (Pierre), *Expressions Régulières et Parsing*, 2021, URL : https://python.sdv.univ-paris-diderot.fr/16_expressions_regulieres/ (visité le 21/08/2022).
- GABAY (Simon), *Cours Sur l'OCR et GROBID*, 2020, URL : https://github.com/gabays/Cours_2020_01_Strasbourg (visité le 14/08/2022).
- *Formes Numérisées et Détection Unifiée Des Ecritures*, 2021, URL : <https://www.unige.ch/lettres/humanites-numeriques/index.php?cID=188>.
- GABAY (Simon), CAMPS (Jean-Baptiste), PINCHE (Ariane) et CHRISTENSEN (Kelly), « SegmOnto - A Controlled Vocabulary to Describe and Process Historical Textual Sources » (, 2022), p. 19.
- « SegmOnto : Common Vocabulary and Practices for Analysing the Layout of Manuscripts (and More) », dans *Proceedings of the 1st International Workshop on Computational Paleography, IWCP@ICDAR 2021*, Lausanne (Switzerland), 2021 (Lecture Notes in Computer Science), URL : <https://hal.archives-ouvertes.fr/hal-03336528>.
- 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).
- Git*, URL : <https://git-scm.com/>.
- Github*, URL : <https://github.com/>.

IIIIf : International Image Interoperability Framework, URL : <https://iiif.io/> (visité le 26/08/2022).

JANES (Juliette), PINCHE (Ariane), JAHAN (Claire) et GABAY (Simon), « Towards Automatic TEI Encoding via Layout Analysis », dans *Fantastic Future 21, 3rd International Conference on Artificial Intelligence for Librairies, Archives and Museums*, Paris, France, 2021, URL : <https://hal.archives-ouvertes.fr/hal-03527287> (visité le 25/08/2022).

Jupyter Notebook, URL : <https://jupyter.org/>.

KARPINSKI (Romain), LOHANI (Devashish), BELAID (Abdel), KARPINSKI (Romain), LOHANI (Devashish) et BELAID (Abdel), « Metrics for Complete Evaluation of OCR Performance », *IPCV'18 - the 22nd int'l conf on image processing, computer vision, & pattern recognition* (, juill. 2018), p. 8, URL : <https://hal.inria.fr/hal-01981731>.

KHEMAKHEM (Mohamed), *Grobid-Dictionnaries*, 2020, URL : <https://github.com/MedKhem/grobid-dictionaries> (visité le 07/08/2022).

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).

KIESSLING (Benjamin), « Kraken - an Universal Text Recognizer for the Humanities », *Digital Humanities* (, 2019), URL : <https://dev.clariah.nl/files/dh2019/boa/0673.html> (visité le 11/08/2022).

— « A Modular Region and Text Line Layout Analysis System », dans *2020 17th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, 2020, p. 313-318, DOI : 10.1109/ICFHR2020.2020.00064.

KIESSLING (Benjamin), TISSOT (Robin), STOKES (Peter) et STÖKL BEN EZRA (Daniel), « eScriptorium : An Open Source Platform for Historical Document Analysis », dans *2019 International Conference on Document Analysis and Recognition Workshops (ICDARW)*, 2019, t. 2, p. 19-24.

Lxml - Processing XML and HTML with Python, URL : <https://lxml.de/> (visité le 15/07/2022).

Markdownguide.Org, URL : <https://www.markdownguide.org/>.

MITTAGESSEN, *Kraken*, août 2021, URL : <https://github.com/mittagessen/kraken> (visité le 12/08/2022).

NOUSIAINEN (Sami), *Report on File Formats for Hand-Written Text Recognition (HTR) Material : CO :OP Community as Opportunity The Creative Archives' and Users' Network*, National Archives of Finland, 2016, p. 69.

OCR4all, URL : <https://github.com/OCR4all> (visité le 21/07/2022).

- PAGE-XML*, URL : <https://github.com/PRIMa-Research-Lab/PAGE-XML> (visité le 23/07/2022).
- PLETSCHACHER (Stefan) et ANTONACOPOULOS (Apostolos), « The PAGE (Page Analysis and Ground-Truth Elements) Format Framework », dans *2010 20th International Conference on Pattern Recognition*, 2010, p. 257-260, DOI : 10.1109/ICPR.2010.72.
- Python 3.7 Documentation*, URL : <https://docs.python.org/3.7/> (visité le 22/07/2022).
- Regex101*, URL : <https://regex101.com/>.
- REUL (Christian), CHRIST (Dennis), HARTELT (Alexander), BALBACH (Nico), WEHNER (Maximilian), SPRINGMANN (Uwe), WICK (Christoph), GRUNDIG (Christine), BÜTTNER (Andreas) et PUPPE (Frank), « OCR4all—An Open-Source Tool Providing a (Semi-)Automatic OCR Workflow for Historical Paintings », *Applied Sciences*, 9–22 (janv. 2019), p. 4853, DOI : 10.3390/app9224853.
- SCHEITHAUER (Hugo), CHAGUÉ (Alix), GABAY (Simon), ROMARY (Laurent), JANÈS (Juliette) et JAHAN (Claire), « From Page to Content – Which TEI Representation for HTR Output ? », dans *Proceedings of the next Gen TEI, 2021, Virtual*, 2021, URL : <https://hal.archives-ouvertes.fr/hal-03380807>.
- SCRIPTA, *eScriptorium*, URL : <https://gitlab.inria.fr/scripta/escriptorium> (visité le 15/07/2022).
- SegmOnto*, URL : <https://github.com/SegmOnto> (visité le 08/07/2022).
- STOKES (Peter A.), *eScriptorium : Un Outil Pour La Transcription Automatique Des Documents*, Billet, URL : <https://epheenum.hypotheses.org/1412> (visité le 12/07/2022).
- TEI (Consortium), *TEI P5 : Guidelines for Electronic Text Encoding and Interchange*, 2021, URL : <https://zenodo.org/record/5347789#.YTCYsI5KjIU> (visité le 05/07/2022).
- (éd.), *TEI P5 : Guidelines for Electronic Text Encoding and Interchange*, 2007, URL : <http://www.tei-c.org/Guidelines/P5/> (visité le 09/01/2022).
- Transkribus : AI Powered Platform for Handwritten Text Recognition*, URL : <https://readcoop.eu/transkribus/> (visité le 13/07/2022).
- UNIVERSITÉ DE GENÈVE, *Certificat de Spécialisation En Humanités Numériques*, URL : <https://www.unige.ch/lettres/humanites-numeriques/cours-et-seminaires/certificat-de-specialisation> (visité le 05/09/2022).
- VALENTINE (Greta), *Subject & Course Guides : Optical Character Recognition (OCR) - Getting Started : Other Tools*, URL : <https://guides.lib.ku.edu/ocr/other-tools> (visité le 21/07/2022).
- WORLD WIDE WEB CONSORTIUM (W3C), *XSL Transformation Documentation*, URL : <https://www.w3.org/TR/2017/REC-xslt-30-20170608/> (visité le 16/07/2022).

ZRAMDINI (A.) et INGOLD (R.), « Optical Font Recognition Using Typographical Features », *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20–8 (1998), p. 877-882, DOI : 10.1109/34.709616.

Introduction : entre enjeux techniques, scientifiques et pédagogiques

En 2010 s'est tenu le premier THATCamp Paris à l'École des hautes études en sciences sociales, une "non-conférence" sur les humanités numériques proposant des modalités de communication axées sur la collaboration et la spontanéité des participants. En a résulté un texte d'une postérité conséquente, visant à éclaircir les directions possibles d'un domaine épistémologiquement trouble. Bien que les consensus restent à ce jour instables, le "Manifeste des Digital humanities" avançait quelques définitions toujours pertinentes :

"Les digital humanities désignent une transdiscipline, porteuse des méthodes, des dispositifs et des perspectives heuristiques liés au numérique dans le domaine des Sciences humaines et sociales."¹

De même pour certains de ses grands desseins :

"Nous appelons à l'intégration de la culture numérique dans la définition de la culture générale du XXIe siècle [...] Nous lançons un appel pour l'accès libre aux données et aux métadonnées. Celles-ci doivent être documentées et interopérables, autant techniquement que conceptuellement."

Mais outre le débat sur son caractère disciplinaire (transdisciplinaire ou auxiliaire), et au delà de la quête de vocations profondes (méthode, partage, connaissance, etc.), ce texte rend manifeste que le mouvement possède une logique pragmatique de développement. En effet, son point 11 aborde de manière explicite la question de son ancrage institutionnel :

"Nous appelons à l'intégration de formations aux digital humanities au sein des cursus en Sciences humaines et sociales, en Arts et en Lettres. Nous souhaitons également la création de diplômes spécifiques aux digital humanities et le développement de formations professionnelles dédiées. Enfin, nous souhaitons que ces compétences soient prises en compte dans les recrutements et les évolutions de carrière."

Plus d'une décennie plus tard, l'on constate une implantation grandissante des hu-

1. THATCamp Paris, *Manifeste Des Digital Humanities*, 2010, URL : <https://tcp-hypotheses-org.ezpaarse.univ-paris1.fr/318>.

manités numériques en tant qu'ensemble de pratiques, et cela en dépit d'un débat agité sur leur signification scientifique et historique². Les transformations profondes que le numérique opère sur le domaine des sciences humaines et sociales constituent un processus en cours pour lequel il est impossible d'établir des bilans durables. C'est en ce sens que certains de leurs aspects les plus concrets semblent être les seuls repères fiables pour évaluer le devenir du mouvement : processus d'institutionnalisation, explosion de l'offre pédagogique et professionnelle, développement des financements pour la recherche³. Il est important de considérer ces réalités comme porteuses de sens ; un axiome fondationnel de la sociologie des sciences est en effet l'idée selon laquelle une pratique savante est constituée aussi bien par ses principes épistémologiques que par ses modalités institutionnelles et sociales d'existence.⁴

Ancrage du numérique dans la recherche, exploration de pratiques pédagogiques renouvelées, production de données et de connaissances. Autant d'enjeux croisés mais différenciés, constitutifs du stage proposé en 2022 par Béatrice Joyeux-Prunel et Simon Gabay pour le master Technologies numériques appliquées à l'histoire (TNAH) de l'École nationale des chartes (ENC). Financé par l'École Normale Supérieure (ENS) et mené au sein de l'Université de Genève (Unige) pour le projet Artl@s, ce stage a consisté dans le développement d'un programme conçu pour accompagner des étudiants dans l'acquisition de connaissances techniques numériques. Le présent mémoire établi un premier bilan de cette mission, introduit par une réflexion sur les enjeux scientifiques qui lui ont donné sens (aussi bien épistémologiques que disciplinaires et institutionnels).

Le travail a été réalisé sur une chaîne de traitement utilisée pour extraire et structurer des données contenues dans des catalogues d'exposition d'art du XIXe et XXe siècles. L'essentiel de l'activité a consisté dans la complétion d'un script python permettant de consolider cette pipeline en deux étapes. La première repose dans l'utilisation d'eScriptorium, une interface graphique pour le logiciel OCR (Optical Character Recognition) kraken. La deuxième consiste dans l'exécution du script python à travers d'une interface pédagogique sur Jupyter Notebook. La vocation principale de ce pipeline est non

2. Aurélien Berra, « Faire des humanités numériques », dans *Read/Write Book 2 : Une introduction aux humanités numériques*, dir. Pierre Mounier, Marseille, 2012 (Read/Write Book), p. 25-43, URL : <http://books.openedition.org/oep/238> (visité le 25/08/2022), "Au cœur du débat sur les humanités numériques, il y a une question récurrente, et même permanente : celle de la définition. Les digital humanities désignent-elles en propre certaines pratiques, des méthodes, une discipline ? De l'aveu de certains praticiens, le terme constitue une sorte de « signifiant flottant ». La communauté propose des réponses à cette question, mais il me semble problématique d'unifier le domaine hors de références à une école, à une pensée ou à un contexte précis. Une bonne réponse serait historique et tiendrait compte des stratégies institutionnelles."

3. Laurent Tessier et Michaël Bourgatte, « Introduction. Enseigner et apprendre les humanités numériques : des savoirs recomposés ? », *Humanités numériques*-5 (5[2022]), DOI : 10.4000/revuehn.2999.

4. Thomas Kuhn, *The Structure of Scientific Revolutions*, Chicago, 1970 ; Timothy Lenoir, « The Discipline of Nature and the Nature of Disciplines », dans *Instituting Science : The Cultural Production of Scientific Disciplines*, 2022, p. 45-74, DOI : 10.1515/9781503616059-005.

pas la production de données, mais l'introduction d'étudiants en sciences humaines à des technologies fondamentales dans le domaine des humanités numériques. Les logiciels et technologies utilisés (eScriptorium, kraken, XML-TEI, XML-ALTO, python), ouvertes et gratuites, sont conformes aux préconisations de la science ouverte. Au delà, la relative facilité de prise en main de cet outil rend envisageable l'alimentation périodique de la base de données du projet Artl@s, qui ambitionne de cartographier les circulations artistiques des XIXe et XXe siècles à l'échelle globale. Cette base, pour laquelle les catalogues d'exposition sont la source primaire privilégiée, a fait ses preuves dans le domaine de l'histoire de l'art tout en étant enrichie jusqu'à présent avec des tableurs saisis manuellement⁵. Elle compte désormais avec une première chaîne de traitement semi-automatique accessible pour des utilisateurs sans connaissances techniques préalables.

L'ingénieur, le chercheur et l'étudiant face aux humanités numériques

La mission proposée en 2022 poursuit une série de stages effectués par des étudiantes du master TNAH depuis 2019. Centrés autour de la création d'un *workflow* pour encoder automatiquement des catalogues numérisés, ils ont été propulsés par divers acteurs institutionnels et ont abordé plusieurs corpus. En 2019, Lucie Rondeau du Noyer a effectué le premier stage avec Simon Gabay au sein de l'Université de Neuchâtel, et a traité des catalogues de ventes de manuscrits du XIXe siècle pour le projet Katabase. En 2020 et 2021, les stages ont eu pour cadre principal le projet Artl@s mené par Béatrice Joyeux-Prunel, qui a poursuivi la collaboration avec Simon Gabay en proposant d'axer le travail sur des catalogues d'exposition d'art. Les chaînes de traitement développées sont spécifiquement adaptées à ce projet, et l'enjeu en 2022 était d'en faire un outil pédagogique simple et efficace.

Béatrice Joyeux-Prunel, titulaire de la chaire en humanités numériques, et Simon Gabay, Maître assistant, sont en effet responsables du certificat de spécialisation en humanités numériques⁶ de l'université de Genève, et assurent des cours d'initiation en niveau Bachelor et Master. Lors du stage 2022, L'utilité scientifique et la performance technique pour produire des données, bien qu'omniprésents, ont été des horizons relégués à un second plan, à la faveur d'une volonté explicite de contribuer à la transmission d'une culture numérique ample pour des chercheurs en devenir⁷. Il s'agit de cultiver leur autonomie en

5. Barbara Topalov, Simon Gabay, Béatrice Joyeux-Prunel, Laurent Romary et Lucie Rondeau du Noyer, « Automating Artl@s - Extracting Data from Exhibition Catalogues », dans 2020.

6. Université de Genève, *Certificat de Spécialisation En Humanités Numériques*, URL : <https://www.unige.ch/lettres/humanites-numeriques/cours-et-seminaires/certificat-de-specialisation> (visité le 05/09/2022).

7. Lyonel Kaufmann, « L'enseignement des SHS est-il soluble dans les Humanités numériques ? Et inversément. » (5 sept. 2019), URL : <https://orfeo.hepl.ch/handle/20.500.12162/3281> (visité le

proposant des systèmes simples et solides qu'ils puissent investir activement, et dans lesquels ils soient en mesure d'intervenir directement. C'est pour cette raison que le choix de technologies gratuites et virtuellement moins efficaces que les solutions proposées par des acteurs privés s'avère pertinent : les enjeux sont la transparence, la reproductibilité et l'autonomie techniques.

Ainsi, l'activité fondamentalement technicienne de la mission a été ponctuée par des discussions avec les directeurs de stage sur sa signification scientifique, tournant autour de réflexions sur les dynamiques établies entre les divers acteurs concernés par ce type de projets : chercheurs, étudiants, ingénieurs. La collaboration n'étant pas envisagée comme un échange tranché entre des individus confinés à des rôles figés, il s'agissait d'appréhender l'agentivité et les limites de chaque contributeur. Il est attendu que l'accompagnement des chercheurs et des étudiants ait pour corollaire le développement de leurs aptitudes, mais aussi de leur compréhension des horizons épistémologiques ouverts par le numérique : hausse exponentielle des données productibles et/ou traitables, ampliation de la gamme d'échelles d'analyse envisageables, démultiplication des narrativités scientifiques possibles⁸. Quant à l'ingénieur, il faut se poser la question de l'impact que son activité a sur la définition des directions scientifiques entreprises. Loin d'être intellectuellement passif, son travail technique défini le champ des possibles sur le plan épistémologique : les liens du savoir ou de la pensée avec les outils qui les conditionnent sont organiques (cela vaut aussi bien pour la transposition des catalogues vers des supports numériques complexes et polyvalents, que pour leur état préalable de technologie en papier).⁹

Loin d'une relativisation des fonctions, ces éléments peuvent contribuer à définir avec plus de rigueur les dynamiques de travail dans le cadre de configurations scientifiques encore inédites il y a quelques décennies. Les réflexions évoquées ont eu un impact certain sur le livrable final, puisqu'il s'agit d'un outil pédagogique pensé dans le cadre des nouvelles dynamiques à l'œuvre dans le champ de la recherche¹⁰. La question de la marge de manœuvre scientifique de l'ingénieur est d'autant plus pertinente que la formation visée par le stage a une vocation technicienne assumée. Le master TNAH se distingue en effet par l'exclusion de la formation à la recherche ; pourtant, les étudiants sont menés à

25/08/2022).

8. Béatrice Joyeux-Prunel (éd.), *L'Art et la mesure : Histoire de l'art et méthodes quantitatives*, Paris, 2022 (Actes de la recherche à l'ENS), URL : <http://books.openedition.org/editionsulm/8557> (visité le 25/08/2022) ; B. Joyeux-Prunel, « L'histoire de l'art et le quantitatif », *Histoire & mesure*, XXIII-2 (2[2008]), p. 3-34, URL : <http://journals.openedition.org/histoiremesure/3543> (visité le 25/08/2022).

9. Christian Jacob (éd.), *Lieux de savoir. 2 : Les mains de l'intellect*, Paris, 2010 ; Mathieu Hauchecorne, « Les « humanités scientifiques » selon Bruno Latour », *Critique*, 786-11 (2012), p. 933-948, URL : <https://www-cairn-info.proxy.chartes.psl.eu/revue-critique-2012-11-page-933.htm> (visité le 21/11/2020).

10. Elie Allouche, *Humanités numériques et pratiques pédagogiques : journée d'étude (02/10/20)*, Éducation, numérique et recherche, URL : <https://edunumrech.hypotheses.org/2108> (visité le 25/08/2022).

cultiver une compréhension solide de ses enjeux¹¹. Il est indispensable que la médiation offerte par les ingénieurs tienne compte des significations profondes des projets, et qu'ils puissent agir en quelque sorte en tant que "techniciens lettrés".¹².

Vers une histoire globale des circulations artistiques à travers des humanités numériques

Le projet Artl@s¹³ est un exemple remarquable des nouvelles configurations possibles dans les milieux académiques depuis le tournant numérique. Il s'agit de mettre à disposition de la recherche en histoire de l'art des instruments numériques ambitieux, construits par le biais d'une collaboration dirigée entre chercheurs, étudiants et ingénieurs. La multiplicité des profils qui y circulent est éloquente : étudiants de licence/Bachelor, master et doctorat en sciences humaines (principalement en histoire de l'art), doctorants et post-doctorants en histoire de l'art avec des aptitudes numériques poussées, étudiants et stagiaires réalisant des diplômes en ingénierie numérique, chercheurs et informaticiens.¹⁴

Artl@s a été lancé en 2009 par Béatrice Joyeux-Prunel (alors Maitresse de Conférences à l'École normale supérieure de Paris), Catherine Dossin (associate professor, Purdue University) et Léa Saint- Raymond (post-doctorante, ENS). Il articule plusieurs projets numériques liés aux mondialisations artistiques du XIXe et XXe siècles, des événements académiques et artistiques liés (journées d'études, conférences, séminaires, expositions, ateliers), ainsi que la revue *Artl@s Bulletin*¹⁵. BasArt¹⁶ en est le projet principal et se trouve directement visé par les chaînes de traitement développées au cours des stages TNAH 2020, 2021 et 2022. Il s'agit d'une base de données géoréférencée de catalogues d'exposition d'art du XIXe et XXe siècles à l'échelle globale. BasArt est une base PostGIS associée à un système d'Information Géographique (SIG), avec une interface visuelle cartographique et plusieurs méthodes de requête pour interroger ses données. Elle contient plus de 3000 catalogues, 200 villes, 50000 entrées et 110000 œuvres.¹⁷

Crée en 2011, mise en ligne en 2016 puis rendue publique en 2018, BasArt a bénéficié de financements de l'agence Nationale de la Recherche (ANR), de l'université Paris

11. Ma propre situation académique m'a imposé de réfléchir continuellement mon positionnement : je suis étudiant de doctorat en histoire et chargé de cours pour des étudiants de Licence 2 en histoire, en même temps qu'étudiant du master TNAH.

12. A. Berra, « Faire des humanités numériques »...

13. B. Joyeux-Prunel, Catherine Dossin et Léa Saint-Raymond, *Artl@s*, URL : <https://artlas.huma-num.fr/fr/>.

14. La page de l'équipe principale fournit de nombreux exemples : <https://artlas.huma-num.fr/fr/lequipe/>

15. <https://docs.lib.psu.edu/artlas/>

16. *Basart*, URL : <https://artlas.huma-num.fr/fr/bases-en-acces-libre/>.

17. B. Joyeux-Prunel, « ARTL@S : A Spatial and Transnational Art History Origins and Positions of a Research Program », *Artl@s Bulletin*, 1-1 (15 sept. 2012), URL : <https://docs.lib.psu.edu/artlas/vol1/iss1/1>.

Sciences Lettres (PSL) et du LabEx Transfers. Temporellement restreints (de trois à quatre ans en moyenne), ces subventions ont permis de mettre en place une plateforme solide mais ne garantissent ni la mise en veille, ni la pérennité de la base. La finalité pédagogique qui sous-tend la saisie des données, effectuée par des chercheurs associés et des étudiants (manuellement puis avec un pipeline semi-automatique), doit être en partie comprise comme une stratégie d'adaptation aux contraintes financières qui déterminent les directions envisageables. Le choix d'une chaîne de traitement ouverte et accessible, axée sur la transmission d'une culture numérique et non pas sur l'efficacité, prend une dimension éthique dès lors que les contributions doivent s'inscrire pleinement dans le cadre de la formation des étudiants. Il s'agit de privilégier, par dessus la production des données, l'apprentissage des outils fondamentaux en humanités numériques.

BasArt a été imaginé par Béatrice Joyeux-Prunel et poursuit ses recherches ainsi que le constat de l'inexistence d'une base centralisée sur les expositions d'art pour les chercheurs en histoire de l'art. Mme Joyeux-Prunel est l'une des principales théoriciennes des notions de "circulation" et de "transfert" artistiques en France, et a contribué à intégrer l'histoire de l'art au tournant global¹⁸ des disciplines historiques pendant les années 2000¹⁹. Le développement de BasArt rend manifeste qu'il ne s'agit pas uniquement d'un instrument de consultation de données, puisque son enrichissement se fait au gré d'évolutions et de contributions scientifiques multiples. Par exemple, l'intérêt récent de Mme Joyeux-Prunel pour la construction de récits décentrés par rapport aux principales capitales artistiques ("axe Paris-New York")²⁰ est concrétisé par la présence de plus en plus conséquente de catalogues d'expositions latino-américaines²¹. BasArt permet de constater la mobilité des œuvres et des exposants à des échelles régionales et mondiales ; pour cette raison, l'enrichissement de la base est à elle seule une activité scientifiquement productive. L'outil permet en effet, au delà d'une histoire globale de l'art, d'observer des circulations "situées" d'individus et d'objets, en accord avec le tournant spatial de la discipline pendant

18. Pierre-Yves Saunier, « Circulations, connexions et espaces transnationaux », *Geneses* (, 2005), p. 110-126, URL : <https://halshs.archives-ouvertes.fr/halshs-00003886> ; Christopher Alan Bayly, Sven Beckert, Matthew Connelly, Isabel Hofmeyr, Wendy Kozol et Patricia Seed, « AHR Conversation : On Transnational History », *The American Historical Review*, 111–5 (2006), p. 1441-1464, DOI : 10.1086/ahr.111.5.1441 ; James Elkins, *Is Art History Global ?*, New York, 2007.

19. B. Joyeux-Prunel, « Ce que l'approche mondiale fait à l'histoire de l'art », *Romantisme*, 163–1 (2014), p. 63-78 ; B. Joyeux-Prunel et Thomas DaCosta Kaufmann, « Reintroducing Circulations : Historiography and the Project of Global Art History », dans *Circulations in the Global History of Art*, dir. Catherine Dossin, Béatrice Joyeux-Prunel et Thomas DaCosta Kaufmann, 2015, p. 1-22 ; B. Joyeux-Prunel, « Les transferts culturels », *Hypotheses*, 6–1 (2003), p. 149-162, URL : <https://www.cairn.info/revue-hypotheses-2003-1-page-149.htm>.

20. Id., « Provincializing Paris. The Center-Periphery Narrative of Modern Art in Light of Quantitative and Transnational Approaches », *Artl@s Bulletin*, 4–1 (11 juin 2015), URL : <https://docs.lib.purdue.edu/artlas/vol4/iss1/4>.

21. À ce sujet, il est important de mentionner une erreur systématiquement reprise dans les mémoires des stages antérieurs : il ne s'agit en aucun cas d'une approche "décoloniale", mais d'un intérêt envers les méthodes de l'histoire globale, transnationale et connectée. On peut signaler que dans son article de 2014 (*Ibid.*), Joyeux-Prunel critique sévèrement la dimension ouvertement militante des postures postcoloniales et le système-monde qu'elles envisagent.

la décennie précédente²². À terme, le dessein est de consolider un panorama diachronique et cartographique fiable sur des processus concrets de mobilité artistique dans un espace-temps global de deux-cent ans.

La source principale de BasArt réside dans les catalogues d'exposition produits pendant la période étudiée. Ces objets peuvent être définis comme des documents semi-structurées, et sont formés de listes d'items inventoriés et organisés, rassemblés en tant qu'ensembles intellectuellement cohérents ; le mot réfère à ces listes mais aussi au support codex qui les contient²³. Comme d'autres technologies papier de type encyclopédique (dictionnaires, annuaires, bibliographies, atlas, etc.), ils s'articulent autour de l'entrée et de l'item comme unités logiques fondamentales. Dans le cadre du projet Artl@s, leur traitement numérique (transposition vers des encodages XML-TEI et des tableurs CSV) soulève de nombreuses problématiques liées aux gains et aux pertes que ces transformations impliquent. L'éclatement de l'unité documentaire implique-t'il la perte d'informations importantes ? La migration rend-elle le support caduque ? S'affranchit-t'elle de ses limites ? Et perd-elle sur le chemin des qualités épistémiques de cette technologie²⁴ ? Le catalogue dématérialisé intègre un ensemble plus ample et acquiert une plasticité qui descend jusqu'au niveau des items. Les œuvres deviennent des unités dont la mobilité peut être désormais mesurée dans le temps et dans l'espace. La granularité de l'encodage, ajustable²⁵, est ici un atout qui a permis d'adapter les données à un outil aux ambitions globales. Mais l'expérience sensible de l'objet, écartée jusqu'à dans la visualité (la base comporte des tableurs et n'intègre pas les images numérisées²⁶), y est perdue définitivement. Au moins sur le plan de la vocation encyclopédique, la base de données rend le catalogue codex virtuellement caduque. Pourtant, la démultiplication des traitements possibles ne peut se substituer à l'expérience concrète de l'objet source, qui est quand à elle épistémologiquement productive pour le chercheur. C'est cette idée qui sous-tend des travaux (et

22. C. Dossin, « Towards a Spatial (Digital) Art History », *Artl@s Bulletin*, 4–1 (11 juin 2015), URL : <https://docs.lib.psu.edu/artlas/vol4/iss1/1>; T. DaCosta Kaufmann, *Towards a Geography of Art*, Chicago, 2004; Jean Marc Besse, « Approches Spatiales Dans l'histoire Des Sciences et Des Arts », *L'Espace géographique*, 39 (2010), p. 211-224.

23. Alain Chante, « La notion de catalogue : de l'imprimé au numérique », *Culture & Musées*, 21–1 (2013), p. 131-152, DOI : 10.3406/pumus.2013.1735 ; Frédéric Barbier, Thierry Dubois, Yann Sordet et Gabriel de Broglie, *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 de Genève, Editions des Cendres, 2015 (Bibliothèques).

24. Bruno Latour, « Ces réseaux que la raison ignore : laboratoires, bibliothèques, collections », dans *Le Pouvoir des bibliothèques. La mémoire des livres en Occident*, dir. Christian Jacob et Marc Baratin, 1996, URL : https://login.proxy.bib.uottawa.ca/login?url=http://www.numilog.com/bibliotheque/BU-Ottawa/fiche_livre.asp?idprod=51632 (visité le 05/12/2020) ; Lorraine Daston et Peter Galison, *Objectivity*, New York, 2007.

25. Alexandre Gefen, « Les enjeux épistémologiques des humanités numériques », *Socio. La nouvelle revue des sciences sociales*–4 (4[2015]), p. 61-74, DOI : 10.4000/socio.1296.

26. Ce choix est assumé : notre chaîne de traitement permet de produire des découpages précises des images des catalogues, au niveau de l'entrée même. Béatrice Joyeux-Prunel a explicitement indiqué que les produits de cette fonctionnalité ne devaient pas être intégrés à BasArt.

des ateliers) réalisés dans un champ récent de l'histoire des techniques, "l'archéologie expérimentale des médias" (Experimental Media Archeology²⁷). L'imagination historienne ne pourrait se passer d'un contact direct et sensible (visuel mais aussi tactile, olfactif, etc.) aux sources conservées. Une approche sensuellement holistique des catalogues est à même de permettre l'appréhension pleine de leurs découpages des domaines de l'activité humaine.

Le corpus constitué pour développer le livrable du stage Artl@s en 2022 pourrait nous permettre de conclure, finalement, que les réflexions qui ont accompagné notre mission ne se superposent pas à un travail fondamentalement technique, lequel se distingue nettement de la recherche. Il s'agit de 37 catalogues numérisés et préalablement transcrits (automatiquement ou manuellement) sur l'interface OCR eScriptorium en format XML ALTO4. Aucune méthode particulière d'échantillonnage n'a été mise en place, ce qui impose un profond déséquilibre en termes spatiaux et temporels. En somme, ce corpus n'est représentatif d'aucune réalité historique, d'aucun processus culturel, et n'a aucune pertinence justifiable en tant qu'ensemble ou collection. Il dépend des choix des contributeurs, des fichiers numériques mis à disposition par des institutions patrimoniales, et des besoins immédiats du projet. C'est sur le plan technique que l'on peut comprendre toute son utilité : ces catalogues offrent des configurations typographiques variées, lesquelles ont permis de développer un instrument capable de produire des extractions très satisfaisantes et possiblement adaptées à l'immense majorité des cas possibles.

27. Andreas Fickers, « Experimental Media Archaeology : A Plea for New Directions », *Annie van den Oever (ed.). Technē/Technology Researching Cinema and Media Technologies – Their Development, Use, and Impact* (Amsterdam : Amsterdam University Press, 2013), pp. 272-278. (), URL : https://www.academia.edu/5578016/Experimental_Media_Archaeology_A_Plea_for_New_Directions (visité le 09/09/2018).

Première partie

Objectifs scientifiques et défis techniques de la mission

Chapitre 1

Poursuivre un projet

Le stage Artl@s/Imago 2022 s'inscrit dans le cadre d'un projet développé par des stagiaires du master TNAH depuis 2019. La mission se poursuit et se précise au fil des années, dans une double perspective partagée entre leur formation et la construction d'instrument pertinents pour contribuer aux projets scientifiques menés au sein d'Artl@s. Depuis le début, la mission a consisté à imaginer et développer un *Workflow* capable d'encoder automatiquement des catalogues.

Le premier corpus était constitué de catalogues de ventes de manuscrit du XVIIe siècle édités au XIXe siècle. Dès 2020, le projet a été adapté aux objets d'étude de BasArt, une base de données mondiale d'expositions d'art du XIXe et XXe siècles. Cette base était alimenté exclusivement par des données issues de transcriptions manuelles sur des tableurs CSV, saisies par des étudiants et des chercheurs associés au projet. La perspective d'un instrument capable de traiter plus rapidement ces objets apparaissait alors comme scientifiquement et techniquement enrichissante.

Les pipelines développées au fil de ces stages ont servi à produire un corpus d'extractions semi-automatiques issues du travail individuel de chaque stagiaire, mais ces outils n'ont été que très rarement utilisées par d'autres personnes. De manière globale, un fil évolutif a mené sur quatre ans à complexifier les enjeux, à améliorer les instruments et à produire des résultats de plus en plus satisfaisants. Il ne s'agit pourtant pas d'un processus linéaire, puisque les enjeux scientifiques et académiques de chaque stage ont changé. Ce chapitre établit un bilan sommaire du travail effectué depuis 2019, pour après positionner notre mission et de définir ses fondements. Il découle des mémoires de chaque stage, de leurs propres visions des missions antérieures, et de conversations avec Simon Gabay sur l'historique du travail effectué. La section "Stages Artl@s" de la bibliographie comporte les mémoires produits suite à ces missions, et inclut la référence du présent travail à titre d'information.

En 2019, Lucie Randoneau du Noyer a entamé une réflexion sur l'encodage automatique de ces technologies papier et livré une première chaîne de traitement. En 2020, Caroline Corbières a systématisé un *workflow* fonctionnel. En 2021, Juliette Janès à re-

construit le pipeline dans son ensemble et a concrétisé une migration vers des logiciels conformes aux principes de la science ouverte. En 2022, les responsables du stage, Béatrice Joyeux-Prunel et Simon Gabay, ont souhaité approfondir ces horizons avec un script plus performant et adapté à leurs pratiques pédagogiques. Il s'agissait de rendre le pipeline efficace et manipulable par des étudiants et des chercheurs en sciences humaines pour contribuer à leur autonomie numérique en même temps qu'à l'enrichissement de la base de données BasArt. L'instrument développé sera utilisé en son état actuel pour introduire des étudiants aux humanités numériques dès la rentrée 2023.

1.1 Stage 2019 : premières réflexions et chaîne de traitement (Transkribus + Grobid)

Le stage effectué par Lucie Rondeau du Noyer en 2019 à l’Institut de littérature française de l’Université de Neuchâtel (UniNe) a livré une première chaîne de traitement permettant la construction d’une base de données XML présentant 75 catalogues du XIXe siècle de ventes de manuscrits. Son mémoire¹ ancre solidement ce travail dans les courants historiographiques concernés, et présente une première réflexion sur les horizons scientifiques et techniques ouverts.

La mission a été dirigée par Simon Gabay au sein de l’Université de Neuchâtel pour le projet e-ditiones. De manière générale, ce projet vise à signaler tous les manuscrits français du XVII inscrits dans des circuits commerciaux, à produire des encodages TEI offrant des descriptions approfondies, puis des éditions scientifiques électroniques. Le stage a porté spécifiquement sur des catalogues de ventes d’autographes, observatoires privilégiés pour comprendre les processus enquêtés. Il s’agit d’instruments fondamentaux pour éclairer l’histoire de la transmission et de la circulation des manuscrits, pour lesquels les traitement numériques restaient balbutiants et limités. L’objectif de la mission était de pallier, dans ce cas concret, à une tendance forte constatée dans la numérisation d’objets patrimoniaux : le manque de structuration des sources historiques et les limites que cette situation impose à la recherche.

Lucie Rondeau du Noyer envisage dans son mémoire, de manière approfondie, les conséquences scientifiques de sa mission dans le champs historiographique traitant la circulation du patrimoine écrit et la provenance des manuscrits. Le livrable qu’elle propose est censé fonctionner comme un instrument d’authentification d’autographes pour des éditeurs scientifiques, des chercheurs et des marchands. La problématique formulée souligne cet ancrage historiographique :

1. 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 Béatrice Joyeux-Prunel, Paris, Ecole nationale des Chartes, 2019.

”en l’absence d’un corpus constitué et aisément accessible de catalogues de vente de lettres autographes, quelles sont les technologies numériques permettant de faciliter la consultation et l’exploitation de ces sources ? Dans quelle mesure la production de documents encodés et mis à disposition des chercheurs peut-elle servir l’histoire de la circulation de l’écrit aussi bien que les recherches philologiques et ecdotiques ?”

Une partie conséquente du travail s’inscrit dans une démarche intellectuelle qui s’éloigne d’aspects purement techniques. Mais la réflexion, portée à certains moments sur le champ de l’histoire de l’art, dégage des enjeux importants pour les tournants successifs qu’ont pris les stages. Dans le cadre du projet Artl@s, les enjeux hitoriographiques esquissés et les explorations techniques qu’ils impliquent sont les mêmes : utiliser une technologie papier de signalisation d’objets patrimoniaux pour cartographier des mobilités (non seulement marchandes) et des processus culturels larges (transferts, circulations).

Dans le cadre du stage de 2019, l’enjeu ultime semble avoir été la structuration automatique, qui permettrait une exploitation pleine de documents numérisés et mis en ligne, donc accessibles, mais dont les possibilités de traitement scientifique s’avéraient extrêmement limitées. Concrètement, la chaîne de traitement proposée prend pour **input** un catalogue numérisé en **.jpeg** ou en **.pdf** et permet d’obtenir en **output** un document XML-TEI interrogeable via une interface dédiée. Le processus est très loin d’être ”automatique” (c’est avec ce terme qu’il est décrit), mais s’est avéré fonctionnel et a posé des bases solides pour la suite. La mission s’est conclu avec la mise en ligne d’un instrument de recherche contenant soixante-quinze numéros de la *Revue des Autographes, des curiosités de l’histoire et de la biographie*, produite par la famille Chavaray et présentée comme un corpus idéal et représentatif des pratiques marchandes bibliophiles de la deuxième moitié du XIXe siècle.

Deux logiciels se sont retrouvés au centre de la chaîne de traitement : Transkribus et Grobid-dictionaries. Des feuilles de transformation XSL ont été développées pour adapter les fichiers aux traitements successifs du début à la fin. Celles-ci ont permis de garantir la qualité des documents pour en faciliter des corrections manuelles.

Transkribus a été utilisé pour produire des transcriptions automatiques des catalogues imprimés en format **.pdf**. Plus que la fonctionnalité OCR (caractères imprimés), l’implémentation HTR (reconnaissance d’écriture manuscrite) est apparu comme très satisfaisante, avec pour seule limite problématique l’impossibilité de transcrire des emphases typographiques et notamment l’italique, signifiant important dans le cadre des catalogues traités. Par ailleurs, il a été noté dès le début du stage que Transkribus allait éventuellement devenir un logiciel payant. La possibilité d’une migration vers kraken a été suggérée dès cette étape, au moment où Simon Gabay commençait à développer un modèle de reconnaissance kraken pour les caractères imprimés du XIXe siècle.².

2. S. Gabay et Ljudmila Petkovic, *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).

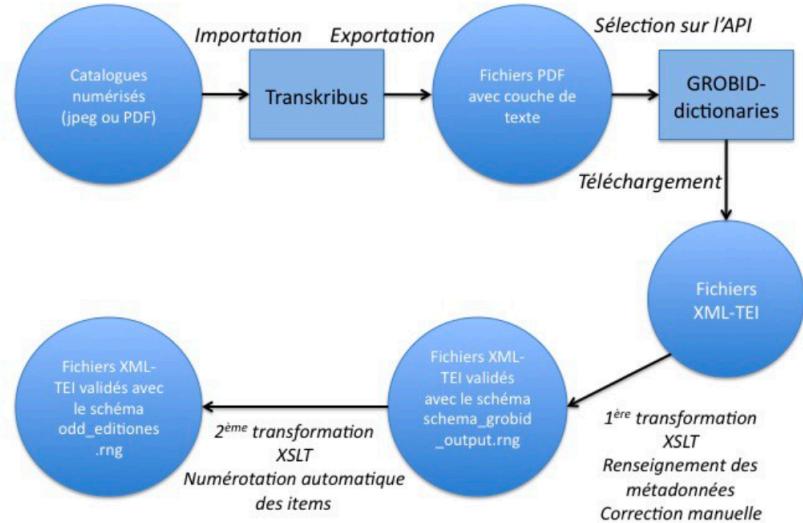


FIGURE 1.1 – Chaîne de traitement en 2019

GROBID-dictionaries est le deuxième logiciel, estimé comme le noyau fondamental de la chaîne. Il s'agit d'une librairie java d'apprentissage supervisé, qui permet d'extraire et de structurer en TEI le texte contenu dans des numérisations de documents de type encyclopédique. Le logiciel prend en entrée les fichiers .pdf et, à partir d'un corpus de données d'entraînement fournies et traitées, applique un modèle de reconnaissance des zones de la page. Il est important de noter que des problèmes techniques importants ont été résolus grâce à la communication avec Mohamed Khemakhem, doctorant en informatique à l'Inria et programmeur du logiciel. Cela a permis d'en faire une brique solide et fonctionnelle de la chaîne de traitement.

Finalement, c'est à travers de feuilles de transformations XSL que les fichiers ont pu traverser chaque étape. En effet, l'output de GROBID n'étant pas conforme au schéma TEI, il fallait remplacer des balises et les préciser sémantiquement. Postérieurement, une deuxième transformation a été utilisée dans l'optique de l'insertion des encodages dans l'architecture du projet e-ditiones.

Le résultat immédiat de cette mission est un livrable permettant la valorisation du corpus traité. Il s'agit d'une base de données XML native permettant de faire des requêtes scientifiquement pertinentes. Le logiciel choisi est eXist-db, qui fournit une interface pour développer des applications Web, la possibilité de construire des index pour les documents, et une implémentation pour la recherche en plein texte. Ainsi, l'application permet de consulter la liste des catalogues, les visualiser, télécharger leurs encodages TEI, et explorer leur contenu avec la recherche plein texte.

Ce premier stage, axé sur le projet scientifique e-ditiones et des réflexions historiographiques poussées, s'est avéré adaptable à une migration vers le projet Artl@s, dont les objets d'étude et les enjeux théoriques restent, du moins structurellement, très similaires.

1.2 Stage 2020 : systématiser un workflow (Transkribus + Grobid + ALTO)

En 2020, un nouveau stage en continuité directe avec l'antérieur a été proposé par Béatrice Joyeux-Prunel au sein du projet Artl@s. En contact avec Simon Gabay et l'équipe ALMAAnaCH de l'INRIA (responsable de développer le logiciel GROBID), elle a proposée la base de données BasArt et le corpus du projet Artl@s comme nouveaux laboratoires pour approfondir la recherche sur l'encodage automatique de catalogues. Il s'agissait donc de développer un outil déployable au sein d'une équipe de chercheurs avec des compétences numériques limitées, adapté à une déclinaison particulière de catalogue fondamentale pour la recherche en histoire de l'art.

La mission a été assurée par Caroline Corbières, dont le mémoire de fin de stage³ retrace une série d'expériences techniques qui l'ont mené à développer un *workflow* adapté au traitement de catalogues d'expositions d'art du XIXe et XXe siècle. Il s'agissait de définir des méthodes adaptées à un corpus vaste et en perpétuel développement, avec le travail de Lucie Rondeau du Noyer comme appui pour automatiser l'encodage d'entrées et les verser dans BasArt. Cette chaîne intègre un nouveau type de document, le fichier ALTO, du fait des horizons qu'il ouvre pour l'encodage et l'extraction du texte sur plusieurs étapes.

Le mémoire de Caroline Corbières s'attarde cette fois non pas sur des enjeux historiographiques, mais sur les implications épistémologiques des réalités institutionnelles à l'œuvre dans l'engrenage scientifique. Les intérêts partagés par plusieurs groupes de travail (Artl@s, e-ditiones, ALMAAnaCH), les atouts et les limites du travail collaboratif, sont présentés comme les fondements ultimes d'un outil qu'il faut situer dans son contexte de production. La mission a été chargée par ces acteurs d'un caractère expérimental, afin de tester des alternatives de plus en plus efficaces dans l'automatisation de l'encodage de catalogues. Ainsi, l'objectif était double : adapter le workflow de Lucie Rondeau du Noyer au projet Artl@s, et évaluer des données inédites prenant en compte des facteurs nouveaux (notamment l'emphase typographique) pour observer le comportement du logiciel GROBID face à ces nouveaux éléments (*features*).

L'apport fondamental de la mission tourne autour d'une réflexion sur l'encodage des catalogues d'art en TEI, mise en pratique avec le développement d'une ODD qui constitue encore à ce jour le schéma des encodages du corpus. Caroline Corbières a décliné des typologies de catalogues d'exposition puis proposé des balises sémantiquement adaptées à une description générale de ces objets, malgré leur hétérogénéité intrinsèque. Des caractéristiques récurrentes ont été signalées et utilisées pour construire un jeu simple mais

3. 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, Paris, Ecole nationale des Chartes, 2020.

(toujours) adapté à tous les catalogues d'exposition traités.

Elle propose une explication de l'arborescence qu'elle a conçue : la liste des entrées de catalogues est encodée dans une balise `<list>`, au sein de laquelle se trouvent toutes les entrées `<entry>`. Tous les catalogues sont composées de deux parties distinctes relevant de l'exposant et de l'œuvre (à l'exception près des expositions individuelles, qui contiennent généralement une liste simple des œuvres). Une `<entry>`, numérotée avec un attribut `@n`, est composée d'une balise `<desc>` (description), où sont indiquées les informations complémentaires relatives à l'exposant, et d'un ou plusieurs `<item>` qui correspondent aux œuvres. Dans ces dernières se trouvent le titre et des informations éventuelles. L'exposant possède aussi une balise `<desc>` est divisée en deux sections :

- `<name>` : nom de l'artiste. Il est possible de différencier nom et prénom avec les balises `<surname>` et `<forename>`
- La balise `<trait>` suivie de `<p>` regroupe les informations complémentaires. Il est possible d'ajouter les balises suivantes : `<birth>`, `<death>`, `<residence>`, `<education>`, `<affiliation>` et `<nationality>`.

L'`<item>` (œuvre), numéroté avec un attribut `@n`, est divisé en trois parties :

- `<num>` : indique le numéro de l'`<item>` tel qu'il apparaît sur le catalogue
- `<title>` : titre de l'œuvre
- `<desc>` : description de l'œuvre. Il est possible d'encoder les informations suivantes : `<material>` (technique artistique), `<dimensions>`, `<date>`, `<history>` (provenance), `<measure>` et `<note>`

La mission a été l'occasion de faire des expériences pour tester des nouvelles conditions techniques dans l'extraction des entrées. La première était l'implémentation prévue du standard XML ALTO sur GROBID, qui a été considérée par les directeurs de stage comme un fichier adapté à une description plus détaillée de la structure typographique des catalogues. En effet, contrairement au format `.pdf`, `input` antérieur pour l'OCR, les fichiers ALTO permettent d'ajouter des indications sur l'emphase typographique. GROBID-dictionaries s'appuie pour sa part sur des *features*, c'est à dire la localisation d'éléments récurrents dans la page, pour établir des modèles de reconnaissance des entrées. Elles avaient été utilisées auparavant pour détecter la mise en page du document selon le positionnement spatial du texte et l'utilisation de la ponctuation, de caractères spéciaux ou de retours à la ligne. Avec ALTO, il est possible d'ajouter de nouvelles *features*, telles que l'emphase typographique avec du gras, l'italique ou la taille de la police. Dans le cadre du travail collaboratif complémentaire au stage, Simon Gabay et Ljudmila Petkovic ont entraîné un modèle HTR qui reconnaît les mots en gras et en italique, puis les encode dans des balises `` et `<i>` grâce à des scripts python.

Cependant, la publication de la version GROBID capable de prendre des fichiers ALTO en `input` a été retardé en raison des nombreuses difficultés techniques que cette

nouvelle fonctionnalité impliquait. Cette contrainte logistique a imposé de changer la stratégie technique : la solution proposée a été un retour vers le format PDF, depuis l’ALTO, qui contiendrait des indications relatives aux emphases typographiques dans le texte même. Ce passage des fichiers ALTO vers du PDF, nouvelle étape, a été mis en œuvre avec XSL-FO, un vocabulaire XSL qui permet de décrire la mise en page d’un document. Postérieurement, il a été possible d’utiliser la nouvelle version de GROBID, et donc d’évaluer la performance des deux méthodes : l’information typographique a eu un impact positif conséquent dans l’extraction des entrées, et fonctionne mieux sur des documents PDF puisqu’il est beaucoup plus complexe de reconstituer le texte à partir des fichiers ALTO.

Outre l’amélioration du processus d’extraction, l’enjeu du stage était sa systématisation, c’est à dire la création d’un workflow réutilisable par les membres du projet d’Artl@s. Il est possible d’observer un schéma de cette chaîne de traitement sur la figure 1.2. Avec un fichier .pdf ou .jpeg en entrée, le *workflow* mènera l’utilisateur à produire un encodage TEI ou un tableau CSV en six étapes :

- import du catalogue (.pdf ou .jpeg)
- OCRisation du catalogue et exportation de la transcription dans des fichiers ALTO avec Transkribus
- transformation des fichiers ALTO en PDF
- encodage automatique avec GROBID-dictionaries
- transformation XSL pour produire un document TEI conforme avec l’output de GROBID
- transformation éventuelle du fichier TEI en tableau CSV

La mission s’est donc achevée avec un *workflow* systématisé. Cependant, certaines limites relatives à son caractère ”User Friendly” sont manifestes et signalés par l’autrice même. Si un guide complet avec des captures d’écran accompagne l’utilisateur dans le déploiement des étapes et des technologies concernées, il est virtuellement impossible d’accomplir le travail sans certaines aptitudes numériques intermédiaires. L’emploi du terminal est inévitable : il faut exécuter des scripts python, lancer des entraînements GROBID, accéder à un serveur local, et connaître les commandes git. Par ailleurs, il s’agit d’une chaîne de traitement hachée, impliquant des technologies complexes et des manipulations intermittentes de fichiers à gérer à chaque étape. Ces conditions allaien sous-tendre des nouvelles directions pour les stages à venir, et notamment la volonté de créer un outil au service des chercheurs et des étudiants.

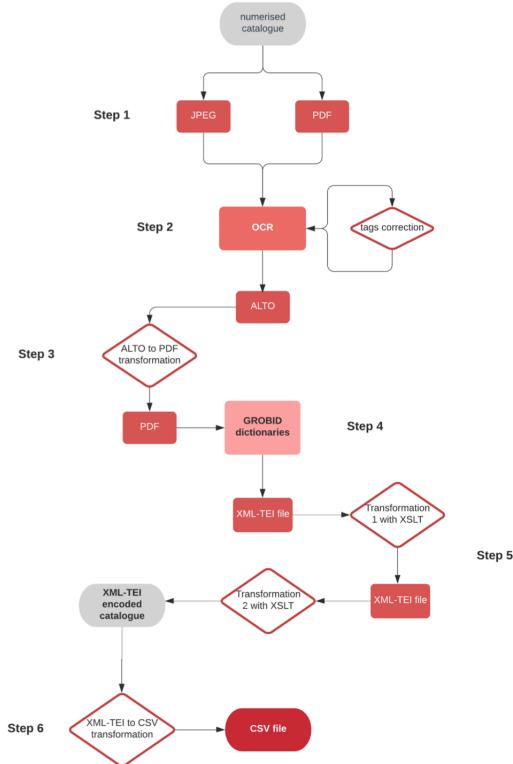


FIGURE 1.2 – Chaîne de traitement en 2020

1.3 Stage 2021 : migration vers la science ouverte (kraken + python)

Un deuxième stage a été proposé en 2021 conjointement par Béatrice Joyeux-Prunel et Simon Gabay. Le logiciel Transkribus est devenu payant en 2020, ce qui a mené à repenser dans ses fondements mêmes la chaîne de traitement développée depuis 2019. Le stage a été donc axé autour d'un réflexion sur la science ouverte et a consisté à concrétiser une migration vers des logiciels libres et gratuits. Juliette Janès a mené cette mission en concentrant ces efforts dans l'entraînement de nouveaux modèles de reconnaissance de caractères. Une deuxième étape a consisté à mettre le logiciel GROBID de côté, pour tester un prototype spécifiquement adapté aux traitement de catalogues d'exposition d'art pour la base de données BasArt. Cette section décrira quelques aspects techniques du script produit à cette occasion, pour pouvoir expliquer postérieurement les transformations que qu'il a subi pendant le stage en 2022.

Le mémoire de fin de stage⁴ de Juliette Janès s'organise autour d'une chronique des transitions opérées, ponctué par une réflexion générale autour des enjeux et des horizons

4. Juliette Janès, *Du Catalogue Papier Au Numérique : Une Chaîne de Traitement Ouverte Pour l'extraction d'informations Issues de Documents Structurés*, mémoire de master « Technologies numériques appliquées à l'histoire », dir. Thibault Cléricie et Béatrice Joyeux-Prunel, Paris, École nationale des chartes, 2021, URL : https://github.com/Juliettejns/Memoire_TNAH.

pragmatiques de la science ouverte. En effet, certaines des préconisations fondamentales⁵ de ce mouvement sont au cœur même du projet Artl@s depuis son inauguration en 2008 : partage de l'information, encouragement de la collaboration scientifique, intégrité et reproductibilité des données et des méthodologies. Béatrice Joyeux-Prunel souhaitait une chaîne de traitement ouverte sans logiciels payants, avec l'objectif spécifique de la rendre reproductible. Le mémoire est axé autour d'un recul critique sur la démarche, avec pour problématique la question pressante de la pertinence d'outils libres mais parfois moins efficaces : "quels sont les apports réels de la restructuration complète d'une chaîne de traitement, pourtant fonctionnelle, dans le but de la rendre intégralement ouverte, libre et gratuite?".

La transformation de Transkribus vers un système de services payants a eu un impact global sur la chaîne de traitement, ce qui a été saisi comme une opportunité pour faire évoluer le projet afin de le rendre plus autonome. Il a rapidement été décidé d'utiliser kraken et l'interface eScriptorium, une application web qui permet de visualiser ses commandes avec des fonctionnalités supplémentaires d'édition manuelle des fichiers. Puis l'utilisation même du logiciel GROBID a commencé à poser certains problèmes. Premièrement, l'équipe ALMAnaCH s'est aperçu que les fichiers ALTO produits par kraken sont différents de ceux produits par Transkribus ; il faudrait attendre à ce que GROBID-dictionaries soit compatible avec l'output de kraken. De plus, le développement du logiciel GROBID-dictionaries semblait compromis du fait du départ de son concepteur de l'INRIA, Mohamed Khemakhem. De plus, malgré son efficacité, il s'agit d'un logiciel qu'il faut entraîner pour chaque catalogue, ce qui complique l'usage du pipeline pour des personnes sans les aptitudes techniques requises. L'idée d'un prototype de script d'extraction des entrées, spécifique aux catalogues de la base de données BasArt, est alors apparue comme un projet souhaitable et envisageable.

La mission s'est concentré sur la migration des jeux de données d'entraînement des modèles d'OCR depuis Transkribus vers kraken. Il s'agit d'un travail complexe qui a été mené de manière très satisfaisante. Une transcription automatique fonctionne avec des modèles entraînés sur des jeux de données particuliers et Transkribus ne permet pas d'extraire ces éléments pour les injecter dans un nouveau logiciel. Il fallait donc procéder à une campagne complète de réentraînement des modèles, avec des données issues des corpus des projets Artl@s et Katabase. Cela a impliqué des efforts considérables, mais plusieurs atouts sont manifestes : il est possible de contrôler et d'améliorer l'OCR, ainsi que de connaître dans son ensemble le fonctionnement du processus de transcription. Il a

5. URFIST Méditerranée, *Les Principes FAIR*, URL : <https://doranum.fr/enjeux-benefices/principes-fair/> (visité le 16/07/2022) ; Mark D. Wilkinson, Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, Jan-Willem Boiten, Luiz Bonino da Silva Santos, Philip E. Bourne, *et al.*, « The FAIR Guiding Principles for Scientific Data Management and Stewardship », *Scientific Data*, 3–1 (mars 2016), DOI : 10.1038/sdata.2016.18 ; Marc Vanholsbeeck, « La Notion de Science Ouverte Dans l'Espace Européen de La Recherche », *Revue française des sciences de l'information et de la communication*–11 (2017), DOI : 10.4000/rfsic.3241.

fallu injecteur dans eScriptorium des fichiers de transcription en `output` de transkribus, adaptés au nouveau logiciel avec des feuilles de transformation XSL. Deuxièmement, des modèles de segmentation et de reconnaissance de caractères ont été entraînés avec ces données. En effet, kraken est un logiciel qui fonctionne sur deux étapes : le *layout analysis* (analyse de mise en page) puis la reconnaissance de caractères à proprement parler. l'OCR repère d'abord la structure de la page et les éléments qui la forment. Il reconnaît aussi les lignes, qui sont matérialisées de deux manières : en tant que *baseline*, ligne suivant le bas du texte, et en tant que masque, qui couvre l'ensemble des caractères de la ligne. Postérieurement, un modèle de reconnaissances de caractères est appliqué.

L'OCR est basé sur du *machine learning*, branche de l'intelligence artificielle qui s'occupe de la création de modèles et d'algorithmes à partir de corpus de données. Il faut fournir des données suffisantes pour que la machine les analyse et apprenne à reconnaître dans l'image des structures de pixels à associer avec des caractères encodés. Il s'agit concrètement de fournir au logiciel des images de page accompagnées de leurs transcription ALTO, ce qui a été fait avec un corpus de 500 pages de catalogues d'exposition (150) et de ventes de manuscrits autographes (350) des XIXe et XXe siècles. Ces données connues et vérifiées par la machine constituent le *grountruth* ("vérité de terrain"). Contrairement à la grande majorité des logiciels OCR, kraken permet la récupération des modèles, ce qui rend possible le partage et la conservation des entraînements.

L'autre volet du stage a consisté à développer un prototype de script python pour extraire les données de l'`output` d'eScriptorium et les structurer en tant que fichiers TEI conformes à l'ODD définie par Caroline Corbières en 2020. Cet outil, placé au centre du stage en 2022, avait été alors conçu comme un instrument expérimental ne pouvant fonctionner qu'avec une sélection restreinte de catalogues. Nous décrirons deux éléments constitutifs de ce programme qui on fait l'objet d'améliorations qui seront présentées postérieurement dans notre mémoire.

Le premier est l'implémentation d'expressions régulières utilisées pour déterminer avec quelles balises encoder chaque ligne. Elles sont censées permettre à la machine de comprendre si la ligne signale un auteur, des œuvres, des informations complémentaires, ou bien des combinaisons des éléments précédents. Un seul fichier python réunissait une série de regex simples adaptées à des cas différents, et l'utilisateur était censé le manipuler pour activer uniquement celles dont il avait besoin. Cela impliquait, pour chaque exécution du script, de vérifier quelle expression régulière est adaptée au catalogue courant, et d'intervenir sur le code.

Deuxièmement, face aux limites techniques impliquées par la perte de l'encodage automatique offert par GROBID-dictionaries, il a été nécessaire de réfléchir à de nouvelles méthodes pour l'extraction textuelle des transcriptions. Celle-ci a mené à définir une typologie de catalogues d'exposition d'art nécessitant des traitements différents et adaptés à chaque cas (voir figure 1.3). Des commandes spéciales ont été créées pour que

l'utilisateur indique le type après une analyse visuelle de la structure des catalogues :

- Simple : pour les catalogues ayant des entrées de type 1.A.
- Double : pour les catalogues ayant des entrées de type 1.B
- Triple : pour les catalogues ayant des entrées de type 1.B et des lignes d'informations supplémentaires
- Nulle : pour les catalogues ayant des entrées de type 1.C

Voici le schéma de la commande pour lancer le script :

```
python3 run.py input titre type titre.xml -v -st
```

Voici une explication de chaque élément :

- `python3` : lance le langage de programmation python3
- `run.py` : exécute le fichier python principal
- `input` : chemin vers les fichiers ALTO contenant les transcriptions
- `titre` : titre pour les ID TEI
- `type` : type de catalogue selon la typologie définie
- `titre.xml` : titre pour le fichier TEI
- `-v` : option de vérification de la structure des fichiers ALTO
- `-st` : option de segmentation et transcription automatique avec kraken et des images en `input`

Pour ce prototype, uniquement des catalogues de type "simple" ont été traités, c'est à dire que chaque ligne ne pouvait correspondre qu'à un seul type d'information bien distincte : exposant, informations, œuvres. Cela a permis d'encoder six catalogues préalablement transcrits, mais le script reste inutilisable pour le reste du corpus. Plusieurs problèmes sont à signaler : les indications d'erreurs sont très limitées, les commandes redondantes et compliquées, et l'exécution implique une analyse humaine préalable du catalogue fourni. Le script fonctionnant rarement, et son exécution nécessitant des ajustements chronophages, il est extrêmement difficile de comprendre l'origine des divers problèmes. De plus, les corrections manuelles doivent être réalisées sur les fichiers ALTO en `input` et sur le fichier TEI en `output`, ce qui peut être très difficile pour des débutants. Juliette Janès signale que, contrairement à la vocation des chaînes de traitement antérieures, cet outil complique à l'extrême, voire interdit la prise en main par des néophytes. Le script propose pourtant une méthode intelligente et prometteuse, développée à partir de rien, qui a retenu l'intérêt des directeurs du stage pour la mission à venir.

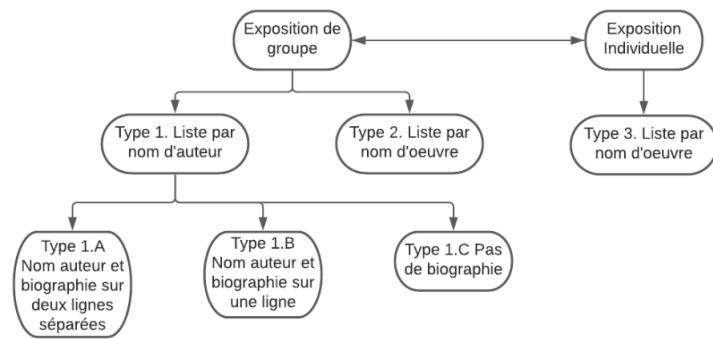


FIGURE 1.3 – Typologie des catalogues pour le programme python en 2021

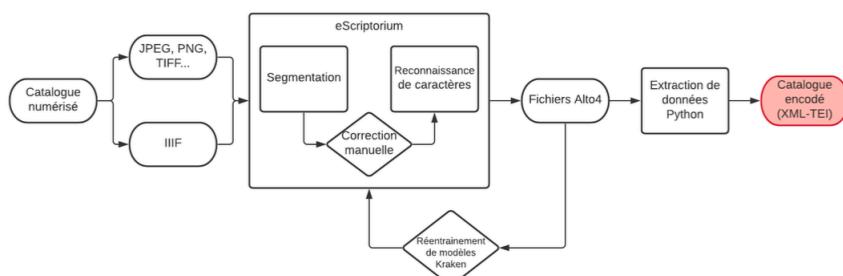


FIGURE 1.4 – Chaîne de traitement en 2021

Chapitre 2

Stage 2022 : produire efficacement des données ou enseigner correctement ?

Ce chapitre ne présente pas un bilan rétrospectif du stage en 2022, qui est l'objet général du présent mémoire. Il tente plutôt, suite à la chronique des missions entre 2019 et 2021, de positionner dans leur continuité l'évolution des enjeux scientifiques, les objectifs des directeurs, ainsi que les problèmes techniques qui ont été affrontés.

Suite à trois stages proposant des explorations complémentaires mais aussi parfois des ruptures radicales, une nouvelle mission pouvait être envisagée avec un horizon pragmatique clair. Les chaînes de traitement possibles, améliorées techniquement et presque totalement conformes aux préconisations épistémologiques et éthiques de la science ouverte, offraient une plateforme solide pour abandonner le terrain prospectif de l'expérimentation. Béatrice Joyeux-Prunel et Simon Gabay souhaitaient d'une part un produit scientifiquement productif, de l'autre un outil pédagogique fiable et abordable pour introduire des étudiants en sciences humaines aux humanités numériques. L'enjeu était de parfaire le pipeline en même temps que de le rendre accessible, dans un contexte où les méthodes employées en vue d'améliorations techniques avaient fini par rendre l'outil de plus en plus difficile à prendre en main. En découlent deux directions tantôt complémentaires, tantôt contradictoires : produire efficacement des données ou enseigner consciencieusement les humanités numériques ?

Nous présenterons la mission dans le contexte des problématiques non résolues par les stages antérieurs, ainsi que dans le cadre de perspectives techniques et scientifiques solides que ces missions ont contribué à construire. Nous évoquerons des directions abandonnées, du fait de la consolidation du sens donné à notre mission, mais aussi du fait de contraintes logistiques qui ont impliqué de mettre de côté certains objectifs. Finalement, nous proposerons une synthèse des contributions techniques effectuées la mission durant,

au regard des problèmes suscités au cours des expériences antérieures.

2.1 Positionnement de la mission

Le stage 2022 s'ancre pleinement dans une continuité avec les missions précédentes, puisqu'il a consisté à reprendre des outils construits antérieurement et présentant des nombreuses problématiques techniques. L'objectif fondamental était de parfaire un pipeline et le rendre pour la première fois véritablement utilisable par des groupes d'utilisateurs (étudiants et chercheurs associés) contribuant à produire des données pour la base BasArt.

Il peut être donc conçu comme un prolongement direct du stage de Juliette Janès, qui a réalisé une migration très satisfaisante vers des instruments conformes à la science ouverte, a entamé un travail approfondi d'entraînement de modèles de reconnaissance sur kraken, et s'est aventuré à proposer un prototype ambitieux pour l'extraction des entrées de catalogue. Afin de constituer une chaîne de traitement authentiquement abordable par des néophytes, il est apparu comme prioritaire de travailler principalement sur cette dernière étape du pipeline. Si cela posait le défi technique de développer un script qui dépasse le statut expérimental, l'horizon envisagé pouvait être considéré comme un nouveau cap franchi pour le projet. Le livrable attendu et réalisé est un pipeline consistant en deux étapes simples à mettre en oeuvre : transcription des catalogues sur l'interface eScriptorium puis exécution d'un script d'extraction des entrées à travers d'un notebook Jupyter.

Outre les limites techniques que l'on retrouve sur le script python développé en 2021, le problème général à résoudre était de remettre au centre du travail une vocation initiale évacuée au fil des stages. En effet, les mémoire successifs prennent soin de noter les difficultés à mettre en place des chaînes de traitement "user friendly". Il est possible d'aller au delà et de dire qu'aucun traitement "automatique" n'a été véritablement atteint (le terme est systématiquement employé pour qualifier les productions). En effet, les logiciels utilisés nécessitent des traitements intermédiaires des fichiers en `input` et en `output`, et il a toujours été nécessaire de faire usage de commandes sur un terminal. Plus est, la transition opérée vers des outils de la science ouverte à bien ouvert les horizons désirés, mais a impliqué de complexifier la chaîne de traitement et de la rendre absolument inabordable par des débutants. Le pipeline actuel se revendique comme semi-automatique, et permet un déploiement immédiat (notamment à l'aide de nombreux exemples qui nécessitent pas de travail préalable sur eScriptorium), en accompagnant tout au long du processus les utilisateurs débutants.

2.2 Projets exclus

Plusieurs projets envisagés avant ou pendant le stage ont été mis de côté, des fois pour des raisons de cohérence technique, mais aussi du fait de contraintes temporelles ; ils peuvent être considérés en ce sens comme des ambitions non satisfaites. Cette section décrit les principaux projets abandonnés en cours de route afin de donner plus d'éléments sur la signification de la mission et de suggérer des améliorations futures.

Premièrement, Simon Gabay avait pensé à développer le script python en confrontant deux méthodes : l'une prolongerait l'extraction commencée par Juliette Janès, qui se base sur la reconnaissance des lignes de texte avec des expressions régulières, et qui inclue une étape préalable qui repère et utilise les zones de la page. Ces zones, conformes au standard de nommage des régions de l'ontologie Segmonto (voir chapitre 6.2.2), sont saisies par l'utilisateur sur l'interface eScriptorium. Il s'agit de signaler directement quel est l'espace occupé par une entrée de catalogue, ce qui a une implication majeure : le script ne peut comprendre que des fichiers ALTO et leur manière d'indiquer ces informations. En conséquence, les seuls fichiers traitables par le programme proviennent de transcriptions contenant des segmentations conformes à l'ontologie Segmonto.

Cela ne pose pas de problème majeur dans le cadre de la vocation pédagogique du pipeline, qui peut ainsi présenter ces trois technologies des humanités numériques à des étudiants. Cependant, l'idée était de créer un deuxième script qui ne prenne pas en compte les régions de la page, et qui fonctionnerait exclusivement avec des expressions régulières. Cela aurait permis, en plus des fichiers ALTO, d'insérer des fichiers de texte brut et .pdf. L'enjeu majeur aurait été alors de savoir s'il était possible qu'un tel programme soit efficace, puisqu'il permettrait alors un enrichissement exponentiel des données de BasArt. Il est donc intéressant d'envisager un tel instrument, qui permettrait de rendre le pipeline plus efficient et automatique, ou bien de confirmer la pertinence d'un travail avec la segmentation des zones. Faute de temps, il a été décidé de concentrer les efforts sur le premier script, afin de le rendre pleinement fonctionnel, puisque ses résultats commençaient à être très satisfaisants et qu'il a été jugé plus important de préparer un instrument pédagogique solide pour la rentrée 2023.

Un autre chemin sans issue a été l'exploration du logiciel de reconnaissance de caractères kraken. Celui-ci avait été mis de côté dès le début dans le cadre du stage, mais son installation en local pour apprendre les manipulations fondamentales paraissait pertinente en vue d'expériences qui pourraient avoir eu lieu en fin de stage. Cependant, l'installation a été extrêmement problématique et nous a fait perdre plusieurs jours de travail, ce qui a rendu impossible de revenir postérieurement sur le logiciel. Il s'agissait d'un conflit de dépendances de librairies python sur MacOS intel 2020 actuellement non résolu, qui a pu être détecté grâce à l'aide de Ben Kiessling, l'ingénieur qui a développé kraken. L'expérience n'était tout de même pas inutile, puisqu'elle nous a fourni un simulacre de travail

collaboratif dans le déboggage de logiciels open source¹.

Une deuxième problématique liée à kraken se trouve dans le script python même, et constitue le seul problème technique à ne pas avoir été résolu pendant la mission. Juliette Janès a intégrée une commande `-st` optionnelle foncièrement expérimentale (puisque non fonctionnelle), qui permettrait de mettre en `input` des images et d'obtenir leur segmentation et transcription automatiques directement à travers du script. Si les problèmes purement techniques du code ont été résolus, cette fonction est inutilisable, car la segmentation automatique n'utilise pas les régions Segmonto nécessaires pour que le script marche correctement. De plus, à ce jour, les modèles de segmentation automatique des zones de la page ne donnent pas de résultats satisfaisants, et notre pipeline implique que l'utilisateur saisisse manuellement ces zones sur l'interface eScriptorium. Ces difficultés pourraient être résolues dans un futur proche : Segmonto est un standard qui se généralise (et Simon Gabay réfléchi à son intégration dans FoNDUE, l'implémentation locale d'eScriptorium de l'Université de Genève) ; deuxièmement, le développement de modèles de segmentation efficaces est au centre de nouveaux stages qu'il dirige. De telles avancées pourraient mener à créer un script python véritablement automatique, nécessitant uniquement des corrections sur les fichiers en `output` (TEI ou CSV).

Nous pouvons mentionner finalement un outil imaginé par Juliette Janès et mis de côté avant le début même du stage 2022. Elle suggérait de créer à partir de son prototype une interface Web dédiée, qui faciliterait l'utilisation du programme python avec des fonctionnalités spécifiques. La proposition était intéressante mais demandait un investissement important de travail. L'idée d'un notebook a paru alors beaucoup plus adaptée, puisqu'il s'agit d'un outil facile à construire et utiliser, déployé au sein même d'un navigateur Web. Une interface dédiée aurait permis tout de même d'aller très loin dans la personnalisation de des outils, et d'implémenter par exemple un instrument pour tester des expressions régulières, ainsi qu'une base d'images de catalogue pour aider l'utilisateur à identifier le type de son document. Cependant, ces fonctionnalités s'avèrent actuellement inutiles, puisque dans son état actuel le script détecte automatiquement le type de catalogue et possède une implémentation regex solide, le tout pouvant être lancé depuis Jupyter Notebook.

2.3 Le choix d'un pipeline échelonné et pédagogique

De manière globale, les projets exclus ont été mis de côté au profit d'un même objectif : développer un pipeline à vocation pédagogique. Béatrice Joyeux-Prunel souhaite que le travail de contribution pour BasArt soit lié à la formation d'étudiants suivant des cours d'introduction aux humanités numériques à l'Université de Genève. Il s'agit

1. L'Issue et sa résolution peuvent être consultés ici : <https://github.com/mittagessen/kraken/issues/348>

d'étudiants réalisant des diplômes en science humaines au niveau bachelor et master, ainsi que du certificat de spécialisation en humanités numériques pour masterants, doctorants et post-doctorants.

Cela permet de solidariser plusieurs missions, telles que la production de données pour la recherche en histoire de l'art et la transmission d'une culture numérique plus solide. Ainsi, le caractère semi-automatique du pipeline devient souhaitable, puisqu'il implique de présenter et de prendre en main des outils ouverts fondamentaux dans le domaine des humanités numériques : OCR (eScriptorium et kraken), python, CSV, et les standards XML TEI et ALTO. Au delà, la méthode étant libre et le code reproductible, son enseignement contribue à la pérennisation du projet, qui peut être récupéré et adapté à de nouveaux usages.

Contrairement à ce qui avait été envisagé lors des stages antérieurs, l'enjeu n'est plus de contourner des lacunes en culture numérique avec un outil automatique, mais de rendre possible l'acquisition d'une autonomie technique fondamentale pour de futurs chercheurs². Il n'y a pas de préalables nécessaires, mais il s'agit de développer des aptitudes et non pas de "bypasser" des obstacles.

Au delà, l'accessibilité peut rendre l'outil efficace, non plus sur le plan purement technique, mais à travers d'une utilisation aisée et donc généralisable. Des groupes d'étudiants vont en faire usage, et la production de données pourra être considéré comme un marqueur pour évaluer le pipeline et mettre en lumière des améliorations nécessaires.

2. Estelle Debouy et Fatiha Idmhand, « Enseigner En Humanités Numériques : Comment Enseigner l'autonomie ? », dans *La Mise En Œuvre Des Humanités Numériques Dans Les Pratiques Pédagogiques En SHS*, Maison de la Recherche de l'université Paris 8, France, 2020, URL : <https://hal.archives-ouvertes.fr/hal-03313016> (visité le 25/08/2022).

Chapitre 3

Résultats du travail

Ce chapitre présente le résultats du travail sur le script extractionCatalogs dans le cadre du stage, de sa poursuite des missions précédentes, et de son insertion dans le projet Artl@s. Il ne fait donc pas une présentation exclusivement technique, et ne fournit pas des explications sur le fonctionnement global et linéaire du pipeline. Pour une approche détaillée, le lecteur doit consulter la partie 3 du présent mémoire. Il peut être préférable de le faire avant dans le cas où le lecteur ne soit pas familiarisé avec les technologies et logiciels évoquées : eScriptorium, kraken, python, XML TEI et ALTO.

Une liste simplifiée indiquant l'ensemble de améliorations apportées au prototype de 2021 de Juliette Janès peut être consultée dans l'annexe A. Celle-ci permet de retracer concrètement les problèmes suscités par le script et les transformations opérées.

Le pipeline comporte deux étapes fondamentales (une liste plus détaillée peut être consultée dans l'introduction de la partie 3) :

1. L'utilisation d'eScriptorium, une interface web qui permet d'utiliser le logiciel d'OCR kraken et qui offre des fonctionnalités supplémentaire d'édition des documents produits.
 - `input` : catalogue numérisé (manifeste `iiif` ou `.jpg`, `.png`, `.pdf`, etc.)
 - Segmentation automatique des lignes
 - Transcription automatique avec un modèle dédié
 - Correction manuelles sur l'interface même
 - Segmentation manuelle sur l'interface même : l'utilisateur doit saisir et nommer les régions sémantiques de la page (entrées, etc.) selon l'ontologie Segmonto
 - `output` : fichiers XML ALTO4 (encodage du texte et de son positionnement spatial dans les zones de la page)
2. L'utilisation du programme extractionCatalogs, qui traite les données produites lors de l'étape antérieure, pour produire les fichiers souhaités (restructurations ALTO, encodage en XML-TEI et tableurs CSV)
 - `input` : fichiers XML ALTO4 produits sur eScriptorium

- lancement du script avec la commande suivante :
- ```
!python3 run.py chemin_input chemin_output nom_du_catalogue
```
- le script python 3 traite automatiquement les fichiers en entrée. Pour chaque fichier ALTO (qui comporte l'encodage d'une page individuelle de catalogue), il analyse le document ligne par ligne pour reconstituer le catalogue, entrée par entrée.
  - **output** : fichier XML-TEI contenant l'encodage des entrées du catalogue.
  - **output** : tableurs CSV produits à partir du fichier TEI. Ceux-ci permettent une lecture humaine et l'injection des données dans la base BasArt
  - **output** : fichier .txt indiquant des problèmes d'extraction, pour que l'utilisateur/utilisatrice puisse faire des corrections manuelles.

### 3.1 Problèmes techniques affrontés pendant la mission

Cette section décrit quelques problèmes techniques affrontés lors de la prise en main du prototype python. Les problèmes d'ordre logistique ou relatifs à des projets sans aboutissement sont décrits dans le chapitre 2.2. cette prise en main a été hasardeuse, du fait de sa complexité technique mais aussi de ses dysfonctionnements. Un premier lancement fonctionnel du programme n'a été possible qu'après certaines corrections sur le code. Le manque de commentaires ou de noms sémantiquement clairs pour les variables et les fonctions ont également ralenti la compréhension du processus. Il a fallu l'appréhender avec la seule lecture des fichiers python .py, faute de résultats satisfaisants en **output**. Voici une liste non exhaustive de problèmes pour le lancement initial :

- Impossibilité de traiter des dossiers contenant des fichiers autres que ALTO (par exemple, leurs images associées)
- Impossibilité de traiter des dossiers avec des fichiers cachés
- Sur MacOs, impossibilité de traiter des dossiers avec des fichiers **.DS\_Store**. Ces fichiers ne peuvent pas être effacés manuellement.
- Fonctions parsemés de *bugs* et arrêt immédiat du script
- Requêtes xpath avec des erreurs (entre autres, non adaptées à la migration vers l'ontologie Segmonto)
- **output** directement déposé dans le dossier du script. L'exécution du script écrase les **output** antérieurs
- lecture désordonnée des fichiers, produisant des fichiers TEI désordonnés et non conformes à l'ODD

Les premières étapes de travail ont consisté à restructurer l'ensemble des dossiers du script, à renommer les fonctions et les variables pour les doter de signification sémantique,

à commenter de manière prolifique et systématique l'ensemble du script, et à organiser un `output` clair dans un dossier spécifique. Il a également fallu corriger le `teiHeader` produit, pour lequel un gabarit a été créé afin de permettre que d'autres projets utilisent le script. L'extraction impliquait de nombreuses erreurs de conformité à l'ODD, qui n'avaient pas été prévues et qui ont pu être réglées sans toucher au schéma : intégration d'items "bis" (répétition des numéros), reconstitutions d'oeuvres ou d'items sur plusieurs lignes, répétition ou erreurs typographiques dans les catalogues, etc.

Cela a permis que le script fonctionne pour des catalogues de type "simple" (les seuls traités par Juliette Janès). Ainsi, il a été possible de commencer à travailler sur un script qui puisse traiter correctement tous les catalogues, qui n'étaient pas reconnus du fait de nombreux *bugs* dans le code. Cela a mené, progressivement, à développer des fonctions capables de reconnaître le type de catalogue en `input` et à adapter leur traitement en fonction des nécessités.

Des discussions avec Frédérine Pradier, étudiante du certificat en humanités numériques travaillant sur des catalogues pictorialistes de la fin du XIXe siècle, ont permis d'avoir les retours d'une utilisatrice du prototype de 2021. Si elle a réussi à adapter légèrement le script à son projet, elle a aussi décrit le *workflow* comme défaillant et entravé. Elle a mentionné des problèmes tels que la difficulté des commandes, les problèmes de numérotation automatique, la non conformité TEI, la non intégration des zones Segmonto `CustomZone:entryEnd`, et la difficulté à comprendre et indiquer le type de catalogue. Par ailleurs l'implémentation d'expressions régulières imposait de les injecter directement dans le code, ce qui était particulièrement difficile à mettre en œuvre. L'encodage de son corpus à donc impliqué une prise en main lente de l'outil, à quoi s'est ajouté la nécessité de corriger manuellement des `output` décevants : problèmes avec les regex ou des entrées particulières, non conformité TEI, etc. La continuité des échanges a permis de comprendre les nécessités d'une utilisatrice modèle et nous a mené à régler un à un les problèmes évoqués. Actuellement, le script `extractionCatalogs` a des résultats très satisfaisants sur les six catalogues du corpus pictorialiste transcrit par Frédérine Pradier.

Un dernier problème majeur du premier script qu'il est important de mentionner est le traitement défaillant des fichiers ALTO en `input` tout au long du programme. Le script n'était pas capable de traiter des fichiers ALTO contenant des balises `TextBlock` vides, qui résultent de fausses manipulations récurrentes dans la segmentation manuelle sur eScriotorium. Aussi, Le premier script était doté d'une commande optionnelle `-v` censée activer une vérification de la conformité des fichiers ALTO, mais celle-ci comportait des erreurs qui arrêtaient directement le programme. Par ailleurs, la propre sortie d'eScriotorium posait des problèmes, puisque l'imbrication des éléments dépend de l'ordre de saisie des zones par l'utilisateur. Le script n'est pas adapté à la multiplicité des cas possibles, ce qui avait pour résultat que seulement 6 catalogues dans un corpus de 37 transcriptions soient traitables. Cette particularité des fichiers ALTO en `output` d'eScriotorium

n'était pas connue de Simon Gabay et il m'a fallu invertir des efforts conséquents pour comprendre la situation. Une discussion avec mes directeurs de stage a mené à déterminer qu'une fonction de restructuration pourrait avoir un intérêt qui dépasserait largement le projet Art1@s.

Nous avons donc traité un à un les problèmes : premièrement, le nouveau script ignore les régions vides de la transcription ALTO, ce qui permet de ne pas générer des entrées vides qui nuisent à la conformité et à la structure de l'`output` TEI (la numérotation résultait inexacte). Deuxièmement, la commande optionnelle `-v` a été corrigée et intégrée au script comme étape constitutive et non optionnelle pour produire des informations utiles sur les documents en `input`.

Par ailleurs, nous avons créé une fonction pour corriger les fichiers ALTO en `output` d'eScriptorium, puisque l'imbrication des éléments dépend de l'ordre de saisie des zones par l'utilisateur et non pas d'une hiérarchie établie selon le degré de précision par rapport au texte. Cette fonction calcule la position des lignes et des zones, et réordonne l'imbrication les éléments (à la manière d'une poupée russe) pour que le script puisse traiter toutes les transcriptions. Cela a rendu possible de traiter les 37 catalogues du script (pour plus de détails, veuillez consulter la section 4.4 de la liste intégrée au chapitre 6.2).

## 3.2 Un instrument efficace et éloquent

Bien au delà de la correction de problèmes techniques, la mission a consisté à doter le script `extractionCatalogs` de fonctions lui permettant de traiter de manière satisfaisante des catalogues d'exposition d'art pouvant comporter des structures assez différentes. L'enjeu a été de créer un outil nécessitant le moins d'indications possibles, et en mesure de donner des retours constructifs en cas de problèmes. Le programme est censé être suffisamment solide pour qu'il n'y ait pas à faire d'interventions sur le code, qui est pour sa part profusément documenté afin de servir à des fins pédagogiques.

Cette section explique certains aspects du fonctionnement antérieur du script qui, sans constituer des erreurs, imposaient des contraintes à dépasser. La commande originale du script demandait, en plus des deux commandes d'exécution python, quatre commandes (dont `type` nécessitant une analyse visuelle préalable du catalogue) et deux options (non fonctionnelles) :

```
python3 run.py input titre type titre.xml -v -st
```

La nouvelle commande, simplifiée, prend uniquement trois indications : le chemin d'entrée, le chemin de sortie souhaité et le titre à donner :

```
python3 run.py input output titre
```

Cette commande permet d'exposer plusieurs améliorations fondamentales. Sa simplicité facilite un prise en main immédiate, appuyée par la présence d'exemples de fichiers

ALTO dans les dossiers `/exemple` du script. Part ailleurs, toutes les contraintes relatives aux chemins sont prises en compte : le dossier en `input` peut contenir des fichiers autres que ALTO, l'on peut choisir le chemin pour l'`output`, lequel est clairement organisé, et il est possible mais non obligatoire d'indiquer la terminaison `.xml` dans le titre du catalogue (avant, il fallait indiquer les deux, avec deux commandes différentes). D'ailleurs, si le titre se termine par `.csv`, le script va produire immédiatement un tableau CSV si un fichier TEI existe déjà, ce qui permet de faire des corrections directement sur lui et d'avoir un CSV en retour sans l'écraser.

Une avancée particulièrement importante est que l'indication du type du catalogue selon la typologie définie par Juliette Janès n'est plus nécessaire. Elle existe toujours au sein du script et permet divers traitements, raison pour laquelle elle constitue un apport fondamental. Mais l'automatisation de la reconnaissance permet un usage extrêmement simplifié du script. En effet, il fallait antérieurement vérifier la structure des entrées puis déterminer son type selon un typologie sémantiquement confuse : "Nulle", "Simple", "Triple", "Dougle" (voir figure 1.3 du chapitre 1.3).

L'autre partie nécessitant des interventions manuelles était l'implémentation d'expressions régulières. Nous avons développé des regex longues et solides adaptées à des cas très variés issues de la totalité des catalogues du corpus (voir chapitre 6.3). Ainsi, aucune intervention n'est requise sur le code pour pouvoir exécuter le programme.

Afin de mieux gérer les erreurs attendues ou possibles, le script est également doté de fonctions de vérification et imprime de nombreux messages d'erreur permettant à l'utilisateur de localiser rapidement des problèmes spécifiques : entrées non intégrées, problèmes de conformité TEI, ALTO ou Segmonto, problèmes dans la production des fichiers en `output`. Une liste complète des messages d'erreur peut-être consultée dans l'annexe D du mémoire. À titre d'exemple, c'est grâce à ces messages que nous avons pu comprendre que les fichiers ALTO en `output` d'eScriptorium comportaient des problèmes d'imbrication. Ces messages, limités au cadre du projet Artl@s, devraient suffire pour faire les corrections nécessaires à toute extraction. Par ailleurs, ces corrections sont facilitées par un fichier de texte brut indiquant toutes les lignes qui n'ont pas été extraites. Nous avons ajouté des indications sur leur nature et sur leur localisation dans l'encodage afin de faciliter la tâche.

Il est important de mentionner que le script est doté d'une sortie claire et concise sur le processus, imprimée sur le terminal sous la forme d'un récapitulatif général de l'extraction. Hormis une liste indentée et visuellement structurée du catalogue dans son ensemble, l'utilisateur obtient un résumé indiquant le nombre d'entrées, d'exposants et d'œuvres extraits.

Finalement, une étape supplémentaire a été ajouté à la chaîne de traitement : la production de deux tableurs CSV, l'un pour lecture humaine, l'autre conçu pour être injecté dans la base BasArt. Ainsi, le pipeline est simplifié en deux grandes étapes, et le

script peut être lancé depuis un notebook Jupyter qui accompagne l'utilisateur dans tout le processus. Il n'est plus nécessaire de faire des manipulations intermédiaires entre ces deux étapes (autre que déposer les fichiers produits par eScriptorium dans un dossier), et l'utilisateur peut être sûr d'obtenir dans tous les cas un `output` utile, qui explicite des problèmes dans les cas où une extraction ne serait pas possible. L'accompagnement de l'utilisateur, ajouté à l'accomplissement d'un programme plus performant, devrait permettre une démultiplication des catalogues traitables. Plusieurs corpus seront proposées aux premiers étudiants qui travailleront avec ce pipeline, dont des manifestes `iiif` de catalogues sur Gallica, des numérisations conservées par l'Université de Pittsburgh, et un corpus du musée d'art et d'histoire de Genève.

### 3.3 Améliorations possibles

La mission s'est terminée avec un livrable utilisable dans un cadre pédagogique. Cependant, ses limites manifestes nous permettent de suggérer des améliorations possibles.

Premièrement, il faut implémenter le traitement de catalogues d'expositions monographiques. Le seul catalogue de ce type dans le corpus<sup>1</sup> donne en effet un `output` défaillant, puisque chaque entrée indique le titre de la première œuvre comme un nom d'auteur. Signalons tout de même que l'extraction est réalisée correctement du début à la fin : il s'agit donc d'une manipulation sur l'organisation TEI de l'extraction sans enjeux techniques particuliers que nous n'avons pas abordé du fait de la courte durée du stage.

Deuxièmement, si le script reconnaît bien les informations complémentaire et possède des expressions régulières adaptées à différent cas (biographies, adresses, prix), celles-ci sont encodées uniquement avec la balise `desc` (description), combien même l'ODD permet de faire ces précisions. Il faudrait donc ajouter les lignes de code nécessaire pour approfondir la précision dans l'encodage des informations complémentaires. Cela impliquera aussi de manipuler les feuilles XSL qui produisent les fichiers CSV, afin que ces précisions soient également intégrées aux tableurs.

Quant au notebook, s'il permet de se passer d'une application Web dédiée, il faut noter qu'il pose une limite : les messages de terminal n'acceptent pas d'`input`, donc certains messages qui posent des questions ne peuvent pas être renseignés (mais ces cas impliquent un usage avancé par des utilisateurs qui devraient normalement lancer le script sur un terminal). Aussi, il serait très intéressant d'intégrer au notebook la possibilité de fournir des regex et de les injecter dans le code, à travers de cellules qui écrivent de nouvelles variables dans le fichier `instanciations_regex.py`. Les cas non prévus par les regex implémentées sont exceptionnels, mais cette fonctionnalité permettrait d'éviter

---

1. Jules-Antoine Castagnary (éd.), *Exposition Des Oeuvres de Gustave Courbet à l'école Des Beaux-Arts (Mai 1882)*, Paris, 1882, URL : <https://gallica.bnf.fr/ark:/12148/bpt6k6574415d?rk=21459;2>.

qu'un étudiant ait à manipuler les fichiers python face à une telle situation.

Finalement, le projet d'extraction directe de fichiers de texte brut ou .pdf pourrait être très enrichissant pour la base BasArt. Il pourrait d'ailleurs être intégré directement au script extractionCatalogs, et tourner automatiquement lorsque l'input n'est pas constitué de fichiers ALTO mais de texte brut ou de fichiers PDF.

Quant aux éléments exogènes au script python, nous pouvons rappeler que la création de modèles de segmentation automatique satisfaisants pourraient rendre la pipeline extrêmement performante. Dans ce cas idéal, la commande expérimentale **-st** pour la transcription et la segmentation automatique pourrait être la clé de voute pour construire un pipeline véritablement automatique. Pour l'instant, aucun objectif particulier est envisagé pour un éventuel stage en 2023. Il faudra attendre de mettre en pratique le pipeline en contexte pédagogique et d'évaluer ses résultats.



# Deuxième partie

## Fonctionnement du pipeline ”extractionCatalogs”



La troisième partie de ce travail présente le pipeline extractionCatalogs dans son ensemble. Celle-ci consiste dans le script python extractionCatalogs, mais aussi dans le traitement préalable des catalogues numérisées avec eScriptorium. L'utilisateur doit aussi prendre en main et installer un notebook Jupyter qui, fonctionnant en tant qu'interface, permet une utilisation plus facile du programme. Ce notebook est en même temps un guide d'utilisation à vocation pédagogique, et présente les méthodes et les technologies concernées à chaque étape. De ce fait, cette troisième partie partage avec le notebook sa structure, qui en constitue une version extrêmement simplifiée.

L'étape d'océrisation avec eScriptorium est fondamentale, et il est nécessaire d'exposer les possibilités, les limites, les bénéfices et les erreurs que cette interface apporte afin d'en faire un usage efficient. Il est cependant important de noter que le stage effectué n'a pas touché à la question de la reconnaissance de caractères, si ce n'est de manière accessoire et superficielle. Le développement de modèles de reconnaissance de caractères et de modèles de segmentation de plus en plus efficaces permettront de renforcer ce pipeline, qui pour le moment nécessite d'interventions manuelles préalables qui peuvent être conséquentes (notamment pour la saisie des régions dans la page, puisqu'à ce jour aucun modèle n'est satisfaisant). Cependant, il faut aussi noter que le script extractionCatalogs, au centre de notre travail, est une étape solide qui fonctionne de manière satisfaisante quel que soit l'origine des transcriptions et des segmentations (automatiques ou manuelles).

Dans cette partie, le chapitre avec le volume le plus important explique le fonctionnement concret et technique du script python extractionCatalogs, dont le développement a été au centre du stage. Il présente les fichiers .py (en langage de programmation python 3) et leur dynamique d'ensemble, ainsi que quelques autres documents mobilisés dans le processus d'extraction des données. Le notebook fournit une très brève explication des objectifs du pipeline dans son ensemble (reconnaissance de caractères + script extractionCatalogs) :

"En partant d'images numérisées de catalogues d'exposition d'art (XIXe et XXe siècle), ce script (programme) nous permet de produire des tableurs (et autres types de fichiers "structurés") organisant toutes leurs informations : exposants, œuvres, informations complémentaires (adresses, entités spatiales, descriptions, associations académiques, prix, etc.)." [...] C'est à dire que nous voulons, à partir d'un catalogue d'exposition sur un support en papier, extraire des informations, les organiser pour signaler leurs significations, et produire ainsi des objets numériques extrêmement utiles pour la recherche en histoire de l'art.

La figure 3.1 montre une page numérisée<sup>2</sup> en `input` (en entrée du pipeline), tandis

---

2. Salon d'automne, Société des artistes décorateurs et Salon des Tuileries, *Catalogue Des Ouvrages*

que la figure 3.2 montre la sortie finale sous forme de tableau CSV. Entre les deux étapes (`input` et `output`) représentées par ces deux figures, il y a un processus semi-automatique avec des interventions manuelles plus ou moins conséquentes selon les cas, mais toujours fondamentales. Nous insisterons sur l'efficacité du pipeline : la dimension manuelle est encadrée et expliquée, et le programme a été conçu pour être fonctionnel avec des utilisateurs néophytes. Ce notebook est en effet un support pédagogique qui sera utilisé dès l'année scolaire 2022-2023 dans les cours d'initiation aux humanités numériques de Béatrice Joyeux-Prunel. Il a vocation à fournir des explications concrètes en cas d'erreur, et dépasse en cette mesure le caractère de prototype qui a pu définir le pipeline les années antérieures. Le notebook ne contient pas uniquement des instructions : il est conçu comme une interface pour lancer le script sans avoir à utiliser un terminal. Il peut donc être utilisé pour effectuer le pipeline, étape par étape.

Comme cela est décrit dans les parties précédentes, les interactions humaines avec les logiciels sont conçues dans (ou adaptées à) un but pédagogique. Les interventions, vérifications et corrections virtuellement nécessaires sont normalement connues et les utilisateurs sont en conséquence censés en être avertis. Si la prise en main de ce pipeline nécessite d'un apprentissage technique (qui en est l'objectif primordial), les résultats acquièrent une performance exponentielle dès lors qu'il s'agit de traiter des corpus conséquents. Rappelons qu'avant la mise en place des stages Artl@s/TNAH, la saisie des tableurs était entièrement manuelle et impliquait donc un investissement temporel extrêmement poussé. Au stade actuel, c'est la saisie manuelle des régions (liée à l'inexistence de modèles de segmentation satisfaisants), puis la correction manuelle des transcriptions automatiques (qui sont de manière générale très efficaces) qui vont demander le plus de temps. Un dernier enjeu, plus explicite lors du stage de Juliette Janès, reste une dimension constitutrice de notre travail : les technologies numériques utilisées sont des outils *Open Source* fondamentaux dans le domaine des humanités numériques. L'objectif de rendre le pipeline conforme aux préconisations pour la science ouverte<sup>3</sup> est repoussé à un nouvel horizon dans la mesure où il s'agit maintenant de former et/ou d'accompagner des chercheurs dans le développement d'une culture numérique solide, susceptible de contribuer à normaliser des échelons supérieurs d'autonomie technique dans l'univers des sciences humaines.

Le pipeline semi-automatique extractionCatalogs (océrisation avec eScriptorium + extraction avec le script extractionCatalogs), accompagné par un guide d'utilisation à vocation pédagogique (Jupyter notebook), a donc un intérêt scientifique (produire des instruments pour la recherche), technique (produire efficacement ces instruments) et pé-

---

*de Peinture, Sculpture, Dessin... Exposés Au Palais de Chaillot... Du 3 Avril Au 25 Avril 1940*, Paris, 1940, URL : <https://gallica.bnf.fr/ark:/12148/bpt6k1179734b/f1.image>, p. 11.

3. UNESCO, *Vers Une Recommandation de l'UNESCO Pour La Science Ouverte*, 2019, URL : [https://en.unesco.org/sites/default/files/open\\_science\\_brochure\\_fr.pdf](https://en.unesco.org/sites/default/files/open_science_brochure_fr.pdf) (visité le 23/08/2022).

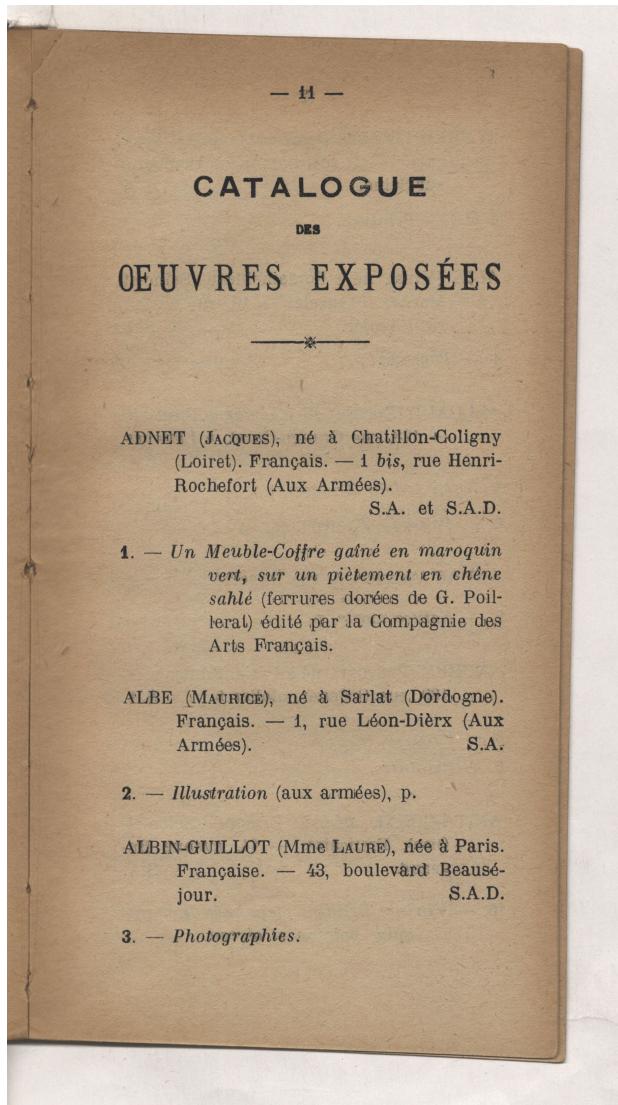


FIGURE 3.1 – **input** : exemple de page de catalogue (Salon d’automne 1940)

dagogique (présenter des technologies numériques fondamentales dans le domaine des humanités numériques). Elle comporte les deux grandes étapes suivantes, dont la description sera approfondie dans les chapitres qui suivent :

1. L'utilisation d'eScriptorium, un logiciel de reconnaissance de caractères (OCR) pour transcrire automatiquement des pages (images) numérisées de catalogue. eScriotorium est une interface qui permet de visualiser toute la chaîne de traitement OCR des images. Ce traitement est en réalité effectué par Kraken, un logiciel qui permet d'entraîner et d'appliquer des modèles de reconnaissance de caractères ainsi que des modèles de segmentation de page
  - **input** : des images numériques de catalogues du XIXe et XXe siècle issues d'un manifeste **iiif** ou en format .jpg, .png, etc.)
  - l'utilisateur/utilisatrice applique un modèle efficace pour que la machine face une transcription automatique des pages

| Last name of the artist          | Biographical elements                                                                                  | Number | Title of artwork                                                          | Subtitle                                                                                                    |
|----------------------------------|--------------------------------------------------------------------------------------------------------|--------|---------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------|
| <b>ADNET (JACQUES)</b>           | , né à Chatillon-Coligny (Loiret). Français. — 1 bis, rue Henri Rochefort (Aux Armées). S.A. et. S.A.D | 1      | Un Meuble-Coffre gainé en maroquin. vert, sur un piétement en chêne sahlé | sur un piétement en chêne sahlé (ferrures dorées de G. Poillerat) édité par la Compagnie des Arts Français. |
| <b>ALBE (MAURICE)</b>            | , né à Sarlat (Dordogne). Français. — 1, rue Léon-Dièrrx (Aux Armées). S.A.                            | 2      | Illustration                                                              | (aux armées), p.                                                                                            |
| <b>ALBIN-GUILLOT (Mme LAURE)</b> | , née à Paris. Française. — 43, boulevard Beauséjour. S.A.D                                            | 3      | Photographies.                                                            |                                                                                                             |

FIGURE 3.2 – output : tableur contenant les informations de la page de catalogue

- l'utilisateur/utilisatrice doit faire des corrections manuelles de la transcription automatique, sur l'interface même
  - l'utilisateur/utilisatrice doit aussi segmenter manuellement ces pages, cet à dire saisir et nommer ses régions sémantiques (il/elle doit signaler où se trouvent les entrées de catalogue dans une image). Dans le projet Artl@s, le choix du nom des régions est conforme a l'ontologie Segmonto (un standard efficace de nommage des zones dans une page)
  - **output** : l'utilisateur/utilisatrice obtiendra en sortie des fichiers XML ALTO4, un type de document adapté à l'encodage des catalogues, puisqu'il tient compte de la localisation spatiale des éléments de la page. Chaque image numérique sera transcrise dans un fichier ALTO ; nous obtiendrons donc un fichier Alto pour chaque page de catalogue
2. L'utilisation du programme extractionCatalogs, qui traite les données produites lors de l'étape antérieure, pour produire les fichiers souhaités (restructurations ALTO, encodage en XML-TEI et tableurs CSV)
- **input** : des fichiers XML ALTO4 produits par l'interface eScriptorium
  - lancement du script avec la commande suivante :
- ```
!python3 run.py    chemin_input    chemin_output    nom_du_catalogue
```
- le script, écrit dans le langage de programmation Python 3, comporte de multiples fonctions qui traitent automatiquement les fichiers en entrée. Pour chaque fichier ALTO (qui comporte l'encodage d'une page individuelle de catalogue), il analyse le document ligne par ligne pour reconstituer le catalogue, entrée par entrée. De manière générale, une entrée est constituée des éléments suivants : nom de l'exposant, informations biographiques/complémentaires, liste des œuvres présentées (avec des éventuelles informations complémentaires)
 - **output** : fichier XML-TEI (encodage structuré) : un seul fichier contenant tout le catalogue et signalant sa structure à partir de balises. Ces balises (par exemple <name>MONET (Claude)</name>) signalent les entrées, les exposants, les œuvres, les informations complémentaires, etc.
 - **output** : tableurs CSV : les mêmes informations sont présentées sous forme de tableaux : un tableau adapté à la base de données BasArt du projet Artl@s, et

qui sera normalement utilisée pour l'alimenter, ainsi qu'une version simplifiée. Cette dernière est conçue pour être comprise aisément par un être humain

- **output** : fichier .txt indiquant des problèmes d'extraction, pour que l'utilisateur/utilisatrice puisse faire des corrections manuelles. Ce document résulte d'un travail approfondi de localisation et de gestion des erreurs attendues (non-conformité des documents en **input**, erreurs de transcription et de segmentation, etc.)

Chapitre 4

Installation : dépôt et Jupyter notebook

Ce chapitre décrit les installations nécessaires pour pouvoir utiliser le script python extractionCatalogs avec le guide/interface Jupyter Notebook. Outre le dépôt avec le programme, il est nécessaire de lancer (ou d'installer et de lancer) Jupyter Notebook ainsi qu'un paquet d'extensions appelé `jupyter_contrib_nbextensions`. Celui-ci facilite une expérience plus "User Friendly" en permettant de visualiser une table des matières avec une structure collapsible. Le guide prévoit qu'un néophyte soit capable de faire ces installations rapidement.

4.1 Installer le dépôt : alternatives

Il est possible de télécharger uniquement le script python, et de l'utiliser en lançant la commande du programme sur un terminal. Cet usage n'est pas recommandé pour des utilisateurs sans expérience, mais il permet de travailler sans la médiation du Notebook et fonctionne de manière autonome. Comme l'indique le fichier `README.md` sur le dépôt Github du projet, il faut utiliser le "Version Control System" Git, qu'il est possible de télécharger pour Mac, Windows et Linux en suivant les indications du site officiel.¹

Git fournit les commandes qui permettront de télécharger depuis Github² le dépôt (projet sous forme d'arborescence de dossiers) du programme extractionCatalogs³. Il faut ouvrir un terminal et lancer les commandes suivantes :

1. Pour la première installation :

- cloner le dépôt :

```
git clone https://github.com/Juliettejns/extractionCatalogs
```

1. *Git*, URL : <https://git-scm.com/>.

2. *Github*, URL : <https://github.com/>.

3. *extractionCatalogs*, URL : <https://github.com/IMAGO-Catalogues-Jjanes/extractionCatalogs>.

- créer un environnement virtuel : `virtualenv -p python3 env`
 - activer l'environnement virtuel : `env/bin/activate`
 - installer les *packages* python nécessaires : `pip install -r requirements.txt`
2. Pour lancer l'application après la première installation :
 - Se déplacer sur le dossier du répertoire installé localement
 - activer l'environnement virtuel : `env/bin/activate`

Cependant, dans le cas d'usage normal à travers du guide Jupyter Notebook, ces installations via le terminal ne sont pas nécessaires. Le guide permet en effet de lancer les commandes en son sein même, ce qui facilite la prise en main pour les étudiants.

4.2 Installer et configurer le guide : Jupyter notebook et nbextensions

Béatrice Joyeux-Prunel et Simon Gabay travaillent avec leurs étudiants du certificat en humanités numériques de l'Université de Genève à travers de Google Colab, un environnement en ligne qui permet d'exécuter du code. Celui-ci est utilisé pour conserver le répertoire et les installations python, qui peuvent dès lors être lancées sur un navigateur web. Il suffit donc que les étudiants aient le lien vers le notebook pour commencer à exécuter les instructions qui suivent. Autrement, il suffirait qu'ils aient en local les fichiers `Guide-extraction-de-catalogues.ipynb` et `requirements.txt`, qu'ils ouvrent un terminal et qu'ils tapent les commandes suivantes :

- aller vers le dossier contenant les fichiers
- créer un environnement virtuel : `virtualenv -p python3 env`
- activer l'environnement virtuel : `env/bin/activate`
- lancer Jupyter notebook (un navaigateur s'ouvrira automatiquement) :

`Jupyter Notebook`

- continuer en suivant les indications du Notebook

Ces manipulations à travers du terminal ne sont pas nécessaires puisque les étudiants peuvent normalement accéder directement au Notebook avec un lien.

4.2.1 Jupyter Notebook

Jupyter Notebook⁴ est un logiciel qui permet d'afficher sur un navigateur des fichiers `.ipynb`, qui présentent des morceaux de code commentés. Cet outil fondamental dans des cadres universitaires (recherche et enseignement) permet donc d'expliquer convenablement

4. *Jupyter Notebook*, URL : <https://jupyter.org/>.

le fonctionnement de scripts. Sur un notebook Jupyter, il existe deux types de cellule : soit pour afficher et exécuter du code, soit pour écrire en MarkDown⁵ (un langage à balise extrêmement simple). Ces dernières sont utilisées pour accompagner le code de commentaires et d'images.

La prise en main de Jupyter Notebook est rapide ; le guide commence en expliquant aux étudiants les manipulations basiques qui leur permettront de lancer le code dans les cellules :

- pour exécuter une cellule de code : taper SHIFT/ENTER, ou bien COMMAND/ENTER sur mac, ou bien le bouton Run sur le menu de la page
- une cellule de code qui commence avec un point d'exclamation (!) n'exécute pas du code, mais une commande sur le terminal.

Il est donc possible d'installer tout le programme et ses *packages* en exécutant une seule cellule. Il suffit pour cela d'avoir le fichier `Guide - extraction de catalogues.ipynb` en local, de l'ouvrir avec un terminal (commande `Jupyter Notebook`) et d'exécuter une cellule contenant les commandes suivantes :

```
!git clone https://github.com/Juliettejns/extractionCatalogs  
!virtualenv -p python3 env  
!pip install -r requirements.txt
```

4.2.2 nbextension : extensions pour Jupyter Notebook

Après les installations fondamentales, le guide invite l'utilisateur à installer le paquet `nbextension`⁶. À nouveau, cela se fait de manière extrêmement simple à travers d'une cellule du Notebook, et permet que celui-ci devienne une interface aisément navigable avec une table des matières interactive et des sections (chapitres et sous-chapitres sur plusieurs niveaux) "collapsables". Les installations se font en exécutant une seule cellule contenant le code suivant :

```
!jupyter contrib nbextension install --user  
!pip install jupyter_nbextensions_configurator  
!jupyter nbextensions_configurator enable --user
```

Ce paquet intègre et active un nouvel onglet sur la page, qui permet d'activer et de désactiver plusieurs add-on (fonctionnalités ajoutées). L'utilisateur doit faire les manipulations suivantes (voir figure 4.1) :

1. cliquez sur le logo "Jupyter" en haut à gauche de la page
2. cliquez sur l'onglet "nbextensions"
3. activez les extensions "collapsible Headings" et "Table of contents (2)"

Configurable nbextensions

disable configuration for nbextensions without explicit compatibility (they may break your notebook environment, but can be useful to show for nbextension development)

filter: by description, section, or tags

<input type="checkbox"/> (some) LaTeX environments for Jupyter	<input type="checkbox"/> 2to3 Converter	<input type="checkbox"/> AddBefore	<input type="checkbox"/> Autoprep8
<input type="checkbox"/> AutoSaveTime	<input type="checkbox"/> Autoscroll	<input type="checkbox"/> Cell Filter	<input type="checkbox"/> Code Font Size
<input type="checkbox"/> Code prettify	<input type="checkbox"/> Codefolding	<input type="checkbox"/> Codefolding in Editor	<input type="checkbox"/> CodeMirror mode extensions
<input checked="" type="checkbox"/> Collapsible Headings	<input type="checkbox"/> Comment/Uncomment Hotkey	<input checked="" type="checkbox"/> contrib_nbextensions_help_item	<input type="checkbox"/> datestamper
<input type="checkbox"/> Equation Auto Numbering	<input type="checkbox"/> ExecuteTime	<input type="checkbox"/> Execution Dependencies	<input type="checkbox"/> Exercise
<input type="checkbox"/> Exercise2	<input type="checkbox"/> Export Embedded HTML	<input type="checkbox"/> Freeze	<input type="checkbox"/> Gist-it
<input type="checkbox"/> Help panel	<input type="checkbox"/> Hide Header	<input type="checkbox"/> Hide input	<input type="checkbox"/> Hide input all
<input type="checkbox"/> Highlight selected word	<input type="checkbox"/> highlighter	<input type="checkbox"/> Hinterland	<input type="checkbox"/> Initialization cells
<input type="checkbox"/> isort formatter	<input type="checkbox"/> jupyter-js-widgets/extension	<input type="checkbox"/> Keyboard shortcut editor	<input type="checkbox"/> Launch QTConsole
<input type="checkbox"/> Limit Output	<input type="checkbox"/> Live Markdown Preview	<input type="checkbox"/> Load TeX macros	<input type="checkbox"/> Move selected cells
<input type="checkbox"/> Navigation-Hotkeys	<input checked="" type="checkbox"/> Nbextensions dashboard tab	<input checked="" type="checkbox"/> Nbextensions edit menu item	<input type="checkbox"/> nbTranslate
<input type="checkbox"/> Notify	<input type="checkbox"/> Printview	<input type="checkbox"/> Python Markdown	<input type="checkbox"/> Rubberband
<input type="checkbox"/> Ruler	<input type="checkbox"/> Ruler in Editor	<input type="checkbox"/> Runtools	<input type="checkbox"/> Scratchpad
<input type="checkbox"/> ScrollDown	<input type="checkbox"/> Select CodeMirror Keymap	<input type="checkbox"/> SKILL Syntax	<input type="checkbox"/> Skip-Traceback
<input type="checkbox"/> Snippets	<input type="checkbox"/> Snippets Menu	<input type="checkbox"/> spellchecker	<input type="checkbox"/> Split Cells Notebook
<input checked="" type="checkbox"/> Table of Contents (2)	<input type="checkbox"/> table_beautifier	<input type="checkbox"/> Toggle all line numbers	<input type="checkbox"/> Tree Filter
<input type="checkbox"/> Variable Inspector	<input type="checkbox"/> zenmode		

FIGURE 4.1 – Activation de la table des matières et des sections collapsables

En cliquant sur ces extensions, l'utilisateur accède à leurs menus, dans lesquels il doit cocher les cases suivantes pour activer des options :

Dans le menu de l'add-on Collapsable Headings :

- "Collapse/uncollapse notebook sections when the ToC2 nbextension is used to collapse/uncollapse sections in the table of contents. For the inverse behaviour, see ToC2's configuration"

Dans le menu de l'add-on Table of Content (2) :

- "Skip h1 headings from numbering, so that they can serve as a notebook title. See the README for details, caveats and alternatives"
- "Collapse/uncollapse ToC sections when the collapsible_headings nbextension is used to collapse/uncollapse sections in the notebook. For the inverse behaviour, see collapsible_headings' configuration"

L'utilisateur peut désormais rafraîchir la page pour accéder à une interface beaucoup plus simple à utiliser (voir figure 4.2) : au lieu d'avoir un long document, il est maintenant possible d'observer les sections et de déplier uniquement celles qu'il a besoin de consulter. Dans le cadre de son travail, l'utilisateur peut ainsi mettre de côté les étapes qu'il a déjà pris en main.

Normalement, c'est après cette installation que l'étudiant lancera une cellule qui installe les packages python du script extractionCatalogs à proprement parler. Les mentions antérieures de commandes sur le terminal (directement ou sur une cellule du notebook avec

5. *Markdownguide.Org*, URL : <https://www.markdownguide.org/>.

6. *Jupyter Notebook Extensions*, URL : <https://jupyter-contrib-nbextensions.readthedocs.io/en/latest/index.html>.

Table des matières ↳

- o Introduction
 - 1 Installation
 - 2 Utilisation du guide : Jupyter notebook
 - 2.1 Code
 - 2.2 GitHub
- o Reconnaissance de texte avec eScriptorium
 - 3 eScriptorium ALTO
 - 3.1 eScriptorium ALTO
 - 3.2 RegEx
 - 3.3 Utiliser le script extractionCatalogs
- o Interventions manuelles et gestion des erreurs
 - 4 Interventions manuelles et gestion des erreurs
 - 4.1 Utiliser les regex
 - 4.1.1 RegEx basiques
 - 4.1.2 RegEx RegeX
 - 4.1.3 RegEx Ocarin
 - 4.1.4 RegEx pour les informations complémentaires
 - 4.2 Messagerie d'erreur et problèmes.txt
 - 4.2.1 erreurs antérieures au script : eScriptorium et input ALTO
 - 4.2.2 Entités non extraites : entry et entryEnd
 - 4.2.3 Entités extraites partiellement
 - 4.2.4 Oeuvres non extraites
 - 4.2.5 Informations complémentaires non extraites

Guide : extraction de catalogues d'art numérisés

ARTIRE / IMAGO / Université de Genève / ENS

Ce guide explique comment utiliser le programme "extractionCatalogs"

En partant d'images numérisées de catalogues d'exposition d'art (XXe et XXIe siècle), ce script (programme) nous permet de produire des tableaux (et autres types de fichiers "structurés") organisant toutes leurs informations : exposants, œuvres, informations complémentaires (adresses, entités spatiales, descriptions, associations académiques, prix, etc.).

- ▶ **0 Introduction** [...]
- ▶ **1 Installation** [...]
- ▶ **2 Reconnaissance de texte avec eScriptorium** [...]
- ▶ **3 Utiliser le script extractionCatalogs** [...]
- ▶ **4 Interventions manuelles et gestion des erreurs** [...]

FIGURE 4.2 – Table des matières du notebook

le signe ”!”) relatives au dépôt ne font pas partie du guide et présentent des manières alternatives de faire les installations. En effet, il suffit pour l'étudiant d'exécuter la cellule suivante pour avoir une installation complète et fonctionnelle (si cela est fait à partir du lien Google Colab, méthode choisie par les responsables du certificat en humanités numériques) :

```
!pip install -r requirements.txt
```

Cette commande installe les paquets(*packages*) python nécessaires pour que le script fonctionne correctement. Les *packages* sont des sous-ensembles du langage de programmation python qui ne sont pas intégrés à la version de base, et qui sont conçus pour être appelés ou installés à la demande des utilisateurs pour des besoins spécifiques.

La cellule affiche alors les messages du terminal, qui donnent des informations sur le processus qui vient d'avoir lieu. Le script va produire des messages qui peuvent être très longs en fonction du catalogue, puisqu'ils signalent toutes les entrées extraites. De ce fait, il est important de rappeler qu'il est possible d'augmenter ou de diminuer la taille d'une cellule en cliquant ou en double-cliquant l'espace en blanc à gauche de celle-ci :

```
In [2]: pip install -r requirements.txt
      nbconvert=>4.2->jupyter_contrib_nbextensions-->r requirements.txt (line 3)) (21.4.0)
Requirement already satisfied: pyrsistent!=0.17.0,!==0.17.1,!==0.17.2,>=0.14.0 in ./env/lib/python3.9/site-packages
(from jsonschema>=2.6->nbformat>=5.1->nbconvert>=4.2->jupyter_contrib_nbextensions-->r requirements.txt (line 3))
(0.18.1)
Requirement already satisfied: wcwidth in ./env/lib/python3.9/site-packages (from prompt-toolkit!=3.0.0,!==3.0.1,<3.1.0,>=2.0.0->python->jupyter-latex-envs>=1.3.8->jupyter_contrib_nbextensions-->r requirements.txt (line 3)) (0.2.5)
Requirement already satisfied: cffi!=1.0.1 in ./env/lib/python3.9/site-packages (from argon2-cffi-bindings->argon2-cffi->notebook)=>0.0-jupyter_contrib_nbextensions-->r requirements.txt (line 3)) (1.15.1)
Requirement already satisfied: executing in ./env/lib/python3.9/site-packages (from stack-data->ipython->jupyter-latex-envs>=1.3.8->jupyter_contrib_nbextensions-->r requirements.txt (line 3)) (0.8.3)
Requirement already satisfied: pure-eval in ./env/lib/python3.9/site-packages (from stack-data->ipython->jupyter-latex-envs>=1.3.8->jupyter_contrib_nbextensions-->r requirements.txt (line 3)) (0.2.2)
Requirement already satisfied: asttokens in ./env/lib/python3.9/site-packages (from stack-data->ipython->jupyter-latex-envs>=1.3.8->jupyter_contrib_nbextensions-->r requirements.txt (line 3)) (2.0.5)
Requirement already satisfied: pycparser in ./env/lib/python3.9/site-packages (from cffi!=1.0.1->argon2-cffi-bindings->argon2-cffi->notebook)=>0.0-jupyter_contrib_nbextensions-->r requirements.txt (line 3)) (2.21)
WARNING: You are using pip version 22.0.4; however, version 22.2.2 is available.
You should consider upgrading via the 'Users/EstebanSanchez/TNHL_Git/IMAG-Artiles/Github_repos/Juliette_Janès/IMAG-0-Catalogues-JJames/extractionCatalogLogs/env/bin/python -m pip install --upgrade pip' command.
```

FIGURE 4.3 – Diminuer une cellule sur le notebook

À présent, le processus d'installation est terminé. L'utilisateur a lancé ses premières commandes sans avoir à utiliser un terminal, et a effectué l'installation de **packages** python sans difficulté. Après cet apprentissage rudimentaire sur l'installation et l'utilisation d'un notebook, il possède une interface qui l'accompagnera à travers de tout le pipeline.

Chapitre 5

Reconnaissance de caractères : interface eScriptorium

L'utilisation de l'interface eScriptorium constitue le début du pipeline à proprement parler. Ce logiciel HTR et OCR fonctionne comme une interface graphique faisant tourner kraken, un programme normalement accessible uniquement à travers d'un terminal `bash`. Pour plus d'informations sur ces technologies, veuillez consultez le chapitre 1.3 du présent mémoire.

eScriptorium permet l'utilisation de kraken par des personnes sans compétences informatiques particulières, d'une manière en principe intuitive et confortable. Il est toutefois important de détailler les consignes à suivre, puisque eScriptorium peut poser quelques contraintes techniques et la démarche à suivre aura un impact dans les fichiers ALTO qu'il produit en `output`. De plus, la prise en main peut paraître évidente pour des usagers habitués ou pour des personnes avec une culture numérique avancée, mais les retours d'étudiants rendent manifeste que l'apprentissage et la compréhension de l'interface ne sont pas immédiats. Pour ces raisons, une description technique et mécanique de l'utilisation préconisée s'avère nécessaire.

5.1 OCR avec eScriptorium et Kraken

eScriptorium permet de transcrire et de segmenter des pages avec du texte, automatiquement aussi bien que manuellement. Cela permet une dynamique fluide entre la machine et l'être humain, puisque la possibilité de faire des interventions et des corrections manuelles est au centre de l'interface. Dans le cas des catalogues d'exposition d'art du XIXe siècle, le modèle de reconnaissance de caractères implanté sur eScriptorium est extrêmement efficace. En effet, le modèle `HTRcatalogs Artlas` a été développé pour le projet Artl@s même. Le résultat de la transcription nécessite tout de même de vérifications et de corrections manuelles plus ou moins importantes selon le catalogue ; une

extraction correcte avec le script python dépend en très grande partie de l'exactitude de la transcription (en particulier les chiffres ou autres caractères permettant de signaler des œuvres, des exposants, ou de séparer les éléments de la page).

La deuxième étape, fondamentale, consiste dans la segmentation des pages, c'est à dire dans la signalisation de ses régions sémantiques (notamment les entrées de catalogue). Les informations de segmentation sont utilisées par le script python pour chercher et reconstituer les entrées, ainsi que pour mettre de côté des informations non pertinentes. Les catalogues d'exposition sont des documents structurés avec des configurations particulières qui permettent de comprendre visuellement la signification des régions contenant du texte, de par leur disposition dans la page ou de par les caractéristiques typographiques du texte. Cette information visuelle, immédiate ou très rapidement appréhendable par l'œil humain, n'est pas discernable par la machine. Kraken permet d'entrainer des modèles de segmentation automatique, mais aucun n'est satisfaisant dans le cadre du corpus traité. En attente de modèles performants, l'utilisateur devra faire la segmentation sur eScriptorium manuellement, page par page. La production de fichiers ALTO avec des régions saisies manuellement pourrait contribuer à terme à entraîner des modèles satisfaisants pour ce projet.

Dans le cadre de cette étape, nous fournirons en `input` des images numériques de catalogues du XIX^e et XX^e siècle. L'interface accepte les principaux types de fichier image, comme `.jpg`, `.png` ou `.pdf`. Il est cependant vivement conseillé d'utiliser un manifeste `iiif`, un standard de partage d'images qui sera présenté dans les pages suivantes. En `output`, eScriptorium fournira un fichier XML-ALTO4 pour chaque image traitée ; ces fichiers seront utilisées lors de la deuxième étape du pipeline (script python `extractionCatalogs`). ALTO est un standard XML choisi dans le cadre du projet Artl@s car il permet d'encoder efficacement la répartition spatiale du texte dans la page. Le script a été conçu pour traiter uniquement ce type de fichier.

Le notebook propose aux utilisateurs de suivre le guide en traitant le catalogue de leur choix. Il expose les étapes à travers d'un exemple représentatif des cas usuels, un catalogue d'une exposition d'art ayant eu lieu à Nancy en 1843¹. Il s'agit d'un catalogue court de 24 pages, dont uniquement 13 avec des entrées à proprement parler. Il sera utilisé aussi comme exemple dans les pages suivantes pour exposer le fonctionnement du pipeline du début à la fin.

Dans le cadre de ce pipeline, nous pouvons distinguer trois étapes dans l'utilisation d'eScriptorium : la mise en place du projet (gestion des paramètres préliminaires, des images et des noms des zones), la transcription automatique du texte contenu dans les images, et la segmentation manuelles des régions contenues dans les images.

1. Société lorraine des amis des arts, *Catalogue Des Peintures, Miniatures, Aquarelles, Dessins, Sculptures et Lithographies Exposés à Nancy... Par Les Artistes Lorrains*, Nancy, 1843, URL : <https://gallica.bnf.fr/ark:/12148/bpt6k62251805.texteImage>.

5.2 Mettre en place un projet sur eScriptorium

En 2022, le stage Artl@s a mis de côté le travail et les réflexions sur la reconnaissance de caractères. En principe, cela dérive d'une division du travail sur les missions de plusieurs stagiaires. Il s'agissait de produire un script d'extraction performant et directement exploitable dans un cadre pédagogique, tandis que le stage de Paul Kervegan s'est penché sur l'entraînement de modèles de segmentation sur kraken avec Simon Gabay. Le présent travail n'aborde donc la question que dans le cadre de l'utilisation de l'interface eScriptorium, et plus spécifiquement des outils et modèles qu'elle met à disposition.

eScriptorium permet de gérer intuitivement toutes les étapes de production de transcriptions, que ce soit à travers d'outils automatiques ou de saisies manuelles. Le projet Artl@s s'ajoute à certains standards de nommage et de saisie des régions, et le script extractionCatalogs nécessite que certains éléments en `input` soient ajustés à son traitement des fichiers ALTO. Pour cette raison, il est fondamental d'expliquer aux utilisateurs comment utiliser eScriptorium de manière extrêmement précise : on s'est rendu compte pendant la mission que l'ordre et les pratiques de saisies ont un impact sur la structure des fichiers en `output`. eScriptorium est une interface en développement constant, et ses fonctionnalités ne peuvent être pleinement adaptées aux cahiers des charges du projet Artl@s. C'est sur la question des segmentations manuelles qu'il faudra être le plus vigilant, mais le script extractionCatalogs permet maintenant de restructurer et de corriger selon nos besoins les fichiers produits par eScriptorium.

On va donc présenter ici, étape par étape, les bonnes pratiques d'utilisation d'eScriptorium et leurs enjeux dans le cadre du projet Artl@s. Les étudiants du certificat en humanités numériques de l'Unige ont accès à FonDue² (FOrmes numérisées et détection unifiée des écritures), une versions d'eScriptorium sur les serveurs locaux de l'université. Il est également possible de suivre les instructions à partir de tout autre installation de l'interface.

5.2.1 Créer un projet

Voici les étapes à suivre pour créer un projet eScriptorium :

- Premièrement, l'utilisateur est invité à créer un projet nouveau en allant sur l'onglet "my projects", et en cliquant sur "Create New Project"



FIGURE 5.1 – eScriptorium : créer un projet

- Il devra choisir un nom pour le catalogue, puis cliquer sur "Create" (figure 5.2)

2. *FoNDUE*, URL : <https://github.com/FoNDUE-HTR/>.

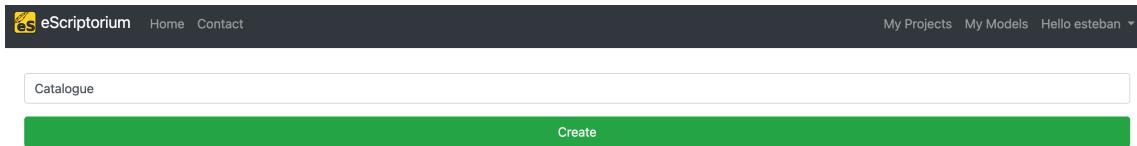


FIGURE 5.2 – eScriptorium : créer un projet (2)

- Il faut ensuite cliquer sur le projet pour accéder à sa page (figure 5.3)

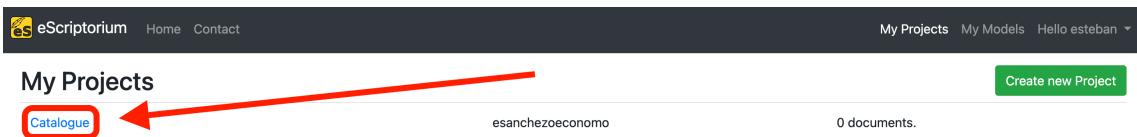


FIGURE 5.3 – eScriptorium : créer un projet (3)

- Puis cliquer sur "Create new Document" (figure 5.4)



FIGURE 5.4 – eScriptorium : créer un projet (4)

- Il faudra ensuite [1] Choisir un nom pour le document et [2] cliquer sur "Create" (figure 5.5)

Ces étapes préliminaires permettent de créer un projet et un document dans ce projet. Les étapes suivantes consistent à établir les noms des zones (cela permettra après de faire la segmentation) puis à ajouter des images.

5.2.2 L’Ontologie Segmonto

Pour signaler les zones qui existent dans chaque page du catalogue, le projet Artl@s préconise d’utiliser l’ontologie Segmonto³. À ce stade, le script a été adaptée à cette ontologie, et en a donc besoin pour pouvoir fonctionner.

Segmonto est un standard de nommage des zones, un vocabulaire contrôlé pour décrire la page manuscrite et imprimée⁴. Son utilisation implique deux avantages immédiats. Premièrement, l’ontologie est issue de discussions et de réflexions menées par des spécialistes en humanités numériques et en philologie, et propose un protocole cohérent et scientifiquement robuste pour nommer les zones d’une page. Deuxièmement, ce standard extrêmement récent est l’une des premières réponses à un intérêt croissant envers

3. *SegmOnto*, URL : <https://github.com/SegmOnto> (visité le 08/07/2022).

4. S. Gabay, Ariane Pinche, Nicola Carboni, Jean-Baptiste Camps, Alexandre Bartz, Cédric Viacoz et Kelly Christensen, « Towards the Fourth Paradigm From Digital Facsimiles of Historical Documents to Highly Annotated Data » (, 2022), p. 14; *Ibid.*

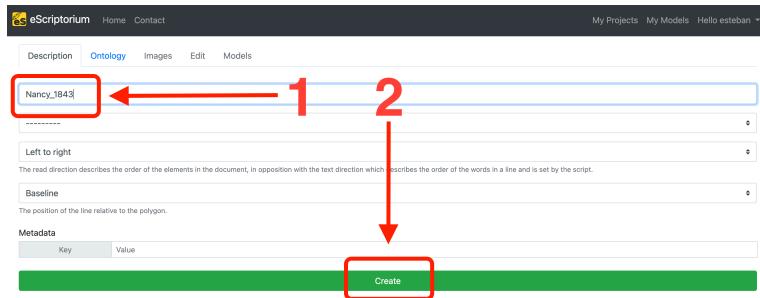


FIGURE 5.5 – eScriptorium : créer un projet (5)

le développement de modèles de segmentation automatiques satisfaisants. Il se généralise rapidement (par exemple, il sera prochainement intégré à l'interface eScriptorium, de manière à que l'utilisateur n'aie plus à saisir les noms manuellement), ce qui renforce la compatibilité du projet Artl@s avec d'autres. Dans ce cadre, le script extractionCatalogs fournit des fonctions qui pourraient elles mêmes s'avérer utiles pour adapter l'output d'eScriptorium à d'autres traitements focalisés sur les régions saisies.

Segmento propose de nombreuses zones de page décrites, et le standard évolue de manière à rajouter des niveaux de précisions adaptés aux besoins de chaque projet. Un ajout récent est la possibilité d'annexer à un type de région standard, après deux points ":", des sous-types et/ou des chiffres mieux adaptés au contexte du projet (par exemple `MainZone:column#1.`).

La liste suivante montre les noms de zone qui ont été sélectionnés dans l'ontologie pour décrire les régions d'une page de catalogue :

- **MainZone** : cette zone fait référence à la totalité de l'espace sémantique de la page. Dans les catalogues Artl@s, elle doit contenir la totalité des entrées et du texte, et exclure des éléments qui ne font pas partie du contenu sémantique de la page, tels que le numéro de page ou des titres courants
- **CustomZone:entry** : CustomZone est un type de zone qui permet à l'utilisateur d'ajouter un nom de son choix ; il indique ainsi un type de zone créé par l'utilisateur. Il était nécessaire de créer une zone adaptée aux entrées de catalogues. une entrée (CustomZone :entry) est composée d'un exposant et des œuvres qu'il expose, avec toutes les informations complémentaires concernées
- **CustomZone:entryEnd** : Ce type d'entrée, plutôt rare, réfère aux cas où la page commence avec la fin d'une entrée dont le début se situe à la fin de la page antérieure (c'est à dire, les cas où une entrée est partagée entre deux pages). Puisque le début de l'entrée commence à la page antérieure, ce type de zone permet que le script extractionCatalogs comprenne qu'il faut les unir, et qu'il ne s'agit pas d'une entrée dans laquelle aucun exposant n'aurait été signalé (cela n'existe pas)
- **NumberingZone** : Ce type de zone encercle les numérotations des pages. Il est très

important de s'assurer qu'elle ne soit pas englobée par la **MainZone**, afin que le script sache qu'il faut mettre de côté cette information qui ne nous intéresse pas dans le cadre de l'extraction. Il n'est pas indispensable de saisir ce type de zone pour que le script fonctionne.

- **GraphicZone:illustration** : Toute image accompagnant le texte. Dans notre cas, il y aura parfois des reproductions d'œuvres d'art insérées dans les pages. Il n'est pas indispensable de saisir ce type de zone pour que le script fonctionne.
- **GraphicZone:ornamentation** : Des ornements courants. Il n'est pas indispensable de saisir ce type de zone pour que le script fonctionne.

La figure 5.6 montre, sur une page de catalogue numérisée, les principaux cas d'usage de zones à signaler.

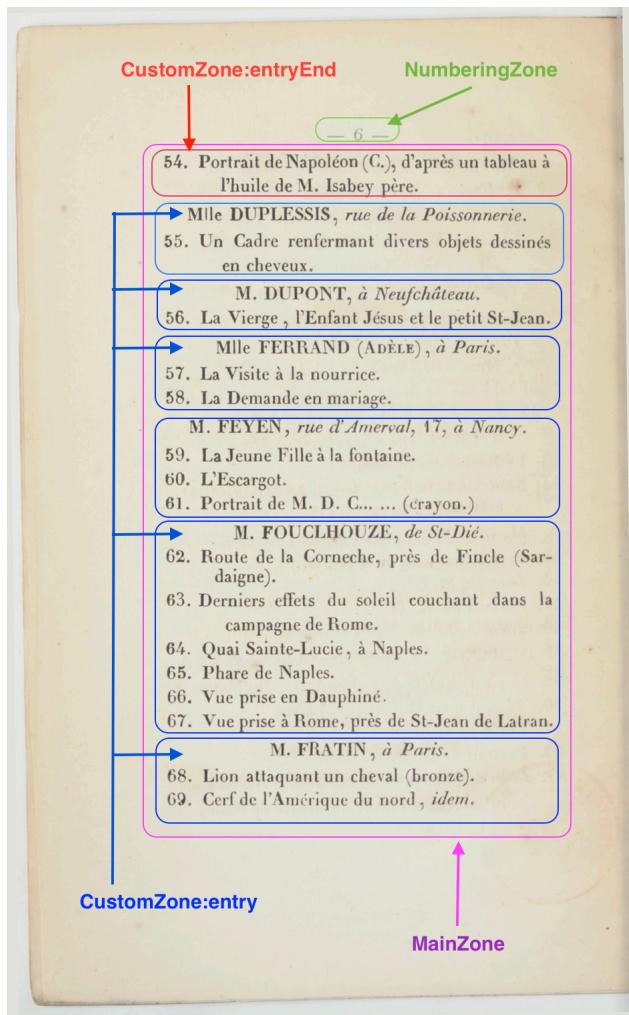


FIGURE 5.6 – Salon de Nancy 1843, p. 6.

Il faut que l'utilisateur prenne en compte et retienne les éléments suivants :

- Les Zones peuvent être parfois serrées, et il est important de veiller à ce qu'elles soient correctement imbriquées ou différencierées : la zone **NumberingZone** est extré-

riéure à la MainZone (espace sémantique), qui elle englobe toutes les CustomZone:entry et une CustomZone:entryEnd.

- Nous avons une CustomZone:entryEnd : c'est à dire une entrée coupée qui commence à la page antérieure (cela est évident dès lors qu'on ne voit pas de personne indiquée). Il est important de signaler ce type de zones pour que le script comprenne qu'il faut l'unir à la dernière entrée de la page antérieure.

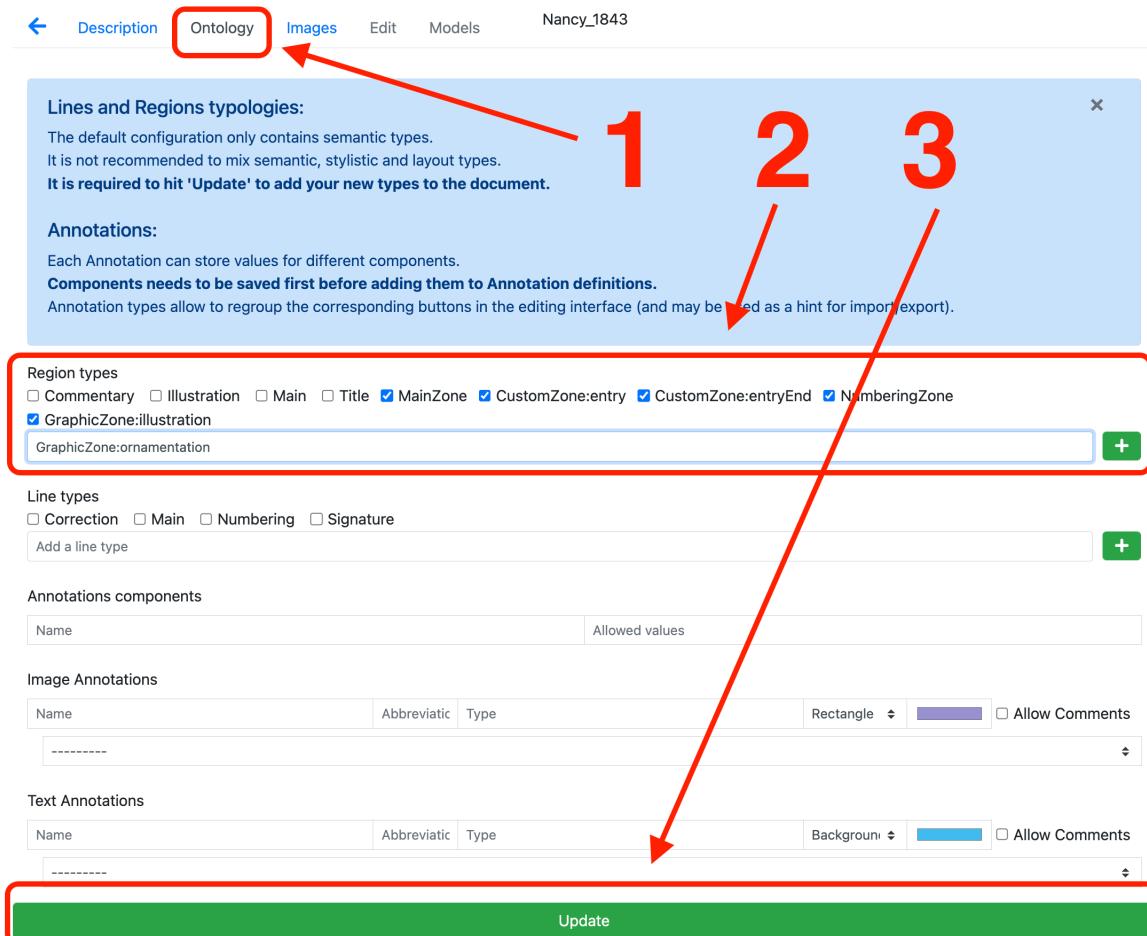


FIGURE 5.7 – eScriptorium : Saisie des zones Segmonto

La figure 5.7 montre les étapes à suivre pour intégrer manuellement l'ontologie Segmonto au projet eScriptorium en cours. Il faut [1] aller sur l'onglet "Ontology" et saisir manuellement chaque zone de la manière suivante :

- copier/coller chacune des zones sur la partie "Region types" et cliquer sur le bouton vert "+" (ajouter) [2]
- saisir une par une les régions, indiquées à continuation :
 - MainZone
 - CustomZone:entry
 - CustomZone:entryEnd
 - NumberingZone

- GraphicZone:illustration
- GraphicZone:ornamentation
- une fois toutes les régions saisies comme sur l'image, Il est très important de ne pas oublier de cliquer sur le bouton **Update** [3]

Une fois cette saisie réalisée, les zones Segmonto pourront être appelées lors de l'étape postérieure de segmentation manuelle.

5.2.3 Quelles images utiliser ? Le standard iiif et les fichiers images classiques (jpeg, png, pdf)

eScriptorium permet d'utiliser les principaux types de fichier image pour réaliser les transcriptions. Il est donc possible d'utiliser des formats communs tels que .jpeg, .png et .pdf. Cela suffira au programme extractionCatalogs pour fonctionner.

Cependant, le projet Artl@s préconise l'utilisation du standard iiif, qui est implémenté également par eScriptorium. iiif⁵ (International Image Interoperability Framework) est un standard de partage et de gestion des images privilégié dans le domaine des humanités numériques. Il permet d'exploiter et de faire circuler efficacement des images numériques de haute qualité entre institutions patrimoniales, universités et chercheurs. La prise en main des outils iiif nécessite un apprentissage technique qui ne sera pas requis dans le cadre de ce pipeline ; mais il est pertinent de mentionner l'existence de cette technologie aux étudiants, et sur un plan purement technique, cela est plus facile pour l'utilisateur.

De plus, lorsque des images iiif sont utilisées, le script extractionCatalogs est capable de les découper, à partir des informations fournies dans le fichier ALTO sur la position des zones. Lorsque le catalogue transcrit sur eScriptorium provient d'un lien iiif, l'extraction sur le terminal intègre un lien vers l'image découpée de chaque entrée. C'est le cas pour le catalogue du Salon de Nancy 1843 donné en exemple, qui ayant été numérisé par la Bibliothèque nationale de France, provient de Gallica et est conforme au standard iiif.

Dans le cadre du pipeline extractionCatalogs, l'utilisation de catalogues conformes au standard iiif est extrêmement simple. Souvent, dans ces cas, le site source fournira l'icône iiif (voir figure 5.8). S'il existe, il devrait permettre de retrouver facilement un lien vers le "manifeste iiif" du document. Celui-ci est un fichier .json contenant des liens vers les images du catalogue ainsi que diverses métadonnées pour le reconstituer dans son ensemble. Dans le cas du catalogue de Nancy 1843, la figure 5.9 montre comment le trouver.



FIGURE 5.8 –
Icône iiif

5. *IIIf : International Image Interoperability Framework, URL : <https://iiif.io/>* (visité le 26/08/2022).

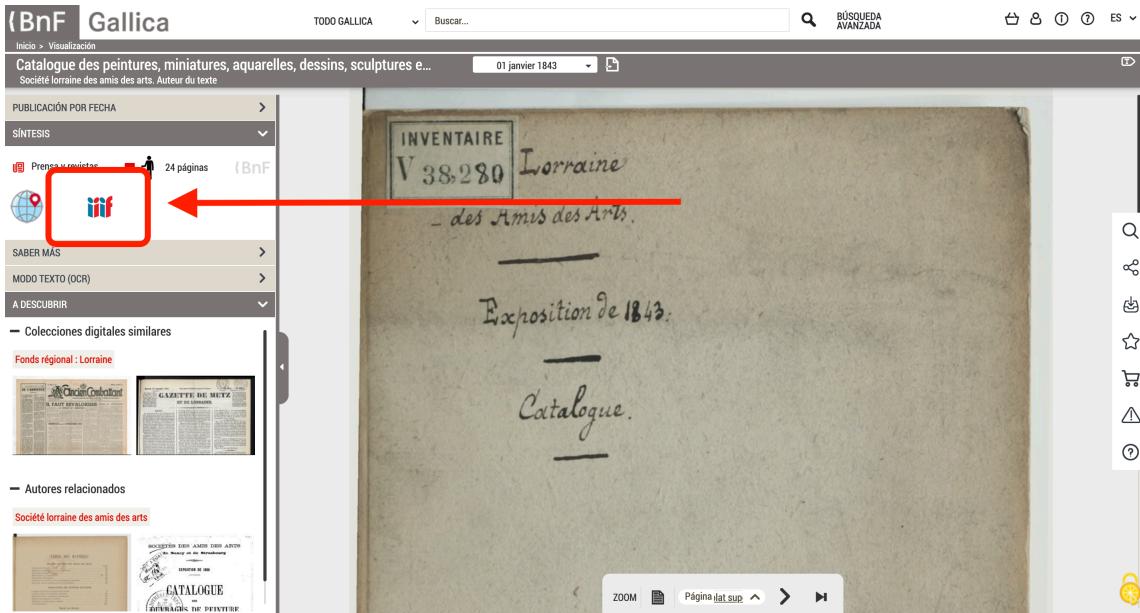


FIGURE 5.9 – iiif sur Gallica

En cliquant sur cette icône, nous accédons à un ”viseur iiif”, c'est un dire un viseur capable de reconstituer un document visuel à travers des informations fournies par un manifeste iiif. Pour récupérer le manifeste iiif qu'il utilise, il faut [1] cliquer sur l d'information, puis [2] scroller jusqu'à la fin, comme indiqué sur la figure 5.10.

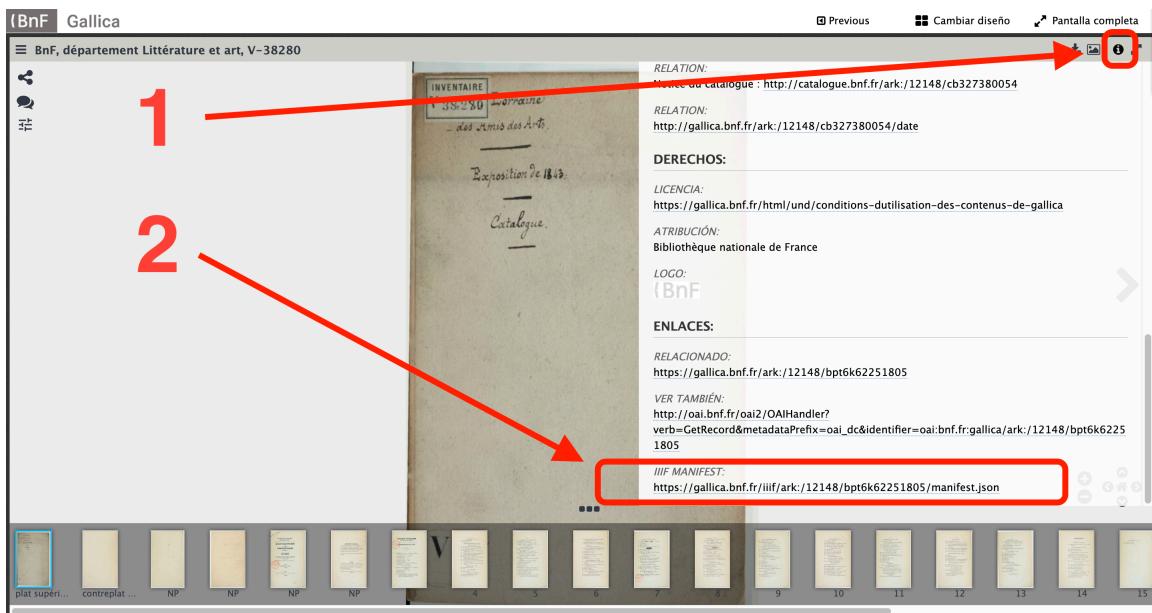


FIGURE 5.10 – Manifeste iiif sur Gallica

On obtient ainsi le lien suivant :

<https://gallica.bnf.fr/iiif/ark:/12148/bpt6k62251805/manifest.json>

Ce lien mène vers le manifeste iiif écrit en langage .json. Ce liens sont faciles à reconnaître car ils se terminent toujours par la chaîne `manifest.json`. Il suffit à l'utilisateur

de copier/coller ce lien sur eScriptorium au moment de saisir les images.

5.2.4 Ajouter des images (iiif, jpeg, png, pdf, etc.)

La figure 5.11 montre comment ajouter les images au document courant sur eScriptorium. Il faut [1] Cliquer sur l'onglet "Images" et choisir une option qui convienne au format des images en `input` :

- 2.1 Déposer des fichiers images une par une (.jpg, .png, etc.)
- 2.2 Cliquer sur Import puis sur IMAGES (IIIF), puis coller un lien vers les manifeste iiif du catalogue. Cliquer sur Start importing : eScriptorium s'occupera d'importer toutes les images
- 2.3 Cliquer sur Import puis sur IMAGES (PDF), puis ajouter un document .pdf Cliquer sur Start importing : eScriptorium s'occupera de le découper en images isolées.

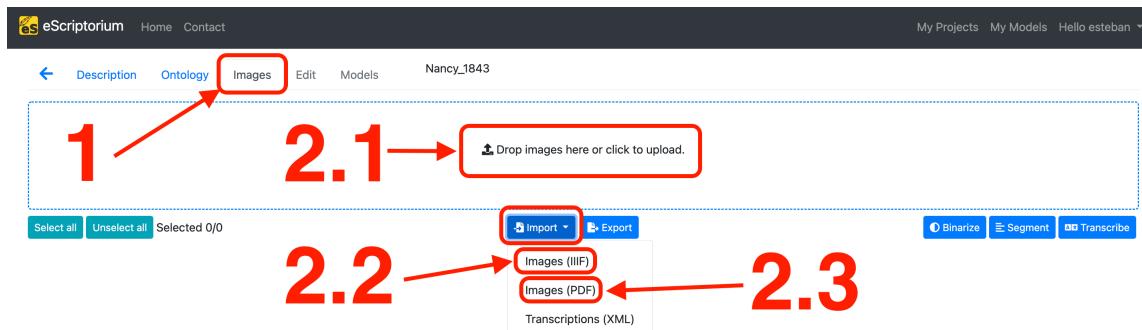


FIGURE 5.11 – eScriptorium : choisir des images

La figure 5.12 montre ce qui se passe dans le cas du catalogue de Nancy 1843 : on obtient une frise avec 24 images.



FIGURE 5.12 – eScriptorium : frise d'images

Lors de l'utilisation du script `extractionCatalogs`, seules les pages contenant des entrées vont être traitées. Cela veut dire que seuls des fichiers qui contiennent des zones segmento `CustomZone:entry` seront prises en compte. L'utilisateur est donc libre de conserver ou d'éliminer les images sans entrées avec la petite croix en haut à gauche de chaque image. Dans notre cas, nous choisissons de garder deux pages sans entrées, afin de montrer explicitement ce qu'il en sera lors du traitement avec le script. La frise comporte désormais 15 images, et les deux premières n'ont aucune entrée.

5.3 Transcription automatique des images

Dans la frise d'images, nous pouvons observer des icônes individuels avec des boutons. La figure 5.13 montre ce niveau et les manipulations fondamentales qu'il permet de réaliser :

1. Case pour sélectionner des ensembles d'images à traiter. Toutes les images dont cette case est cochée subiront les mêmes traitements simultanément.
2. Éditeur d'image : renvoie vers la page d'édition, qui permet entre autres de saisir des transcriptions/segmentations manuelles, ou bien de corriger des transcriptions/segmentations automatiques.
3. Bouton "binariser" : transforme l'image en tons de gris, ce qui facilite la reconnaissance de caractères. Nous avons pas besoin d'utiliser cette fonctionnalité
4. Bouton "segmenter" : permet de segmenter, c'est à dire de reconnaître des régions, mais aussi localiser les lignes dans la page. Dans notre cas, nous ferons usage uniquement de la reconnaissance automatique des lignes.
5. Bouton "transcrire" : permet d'appliquer un modèle de reconnaissance de caractères au choix, selon une liste proposée par eScriptorium.

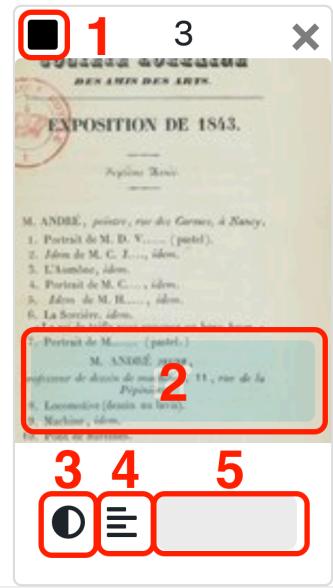


FIGURE 5.13 – eScriptorium : icône image

Il faut d'abord choisir les images à transcrire automatiquement en cochant sur elles. Pour aller plus vite, l'utilisateur peut maintenir les touches COMMAND/SHIFT appuyées et sélectionner la première et la dernière image : toutes les images entre les deux seront cochées automatiquement. Dans le cas du catalogue Nancy 1843, nous allons cocher toutes sauf la première (page de titre), ce qui laisse 14 images, dont une (numéro 2) sans entrées (voir figure 5.14). C'est le cas d'usage pour mettre de côté des images non concernées importées automatiquement à partir d'un manifeste iiif. Maintenant, si on appuie sur

n'importe quel bouton de n'importe quel icône, l'ensemble des images cochées subiront le même traitement.



FIGURE 5.14 – eScriptorium : sélection d'images dans la frise

La première étape consiste à utiliser la fonctionnalité de segmentation automatique, non pas pour reconnaître des régions, mais pour reconnaître des lignes à transcrire. Comme indiqué sur la figure 5.15, il faut [1] cliquer sur n'importe quel bouton "segmenter" d'une des images cochées, ou bien sur le bouton général "Segment". Puis [2] sélectionner l'option "Lines Baselines and Masks" (et surtout pas l'option par défaut "Lines and regions", puisque la segmentation des régions sera faite manuellement). Pour terminer, il faut [3] cliquer sur "Segment". Une fois une image segmentée, son bouton "segmenter" devient vert.

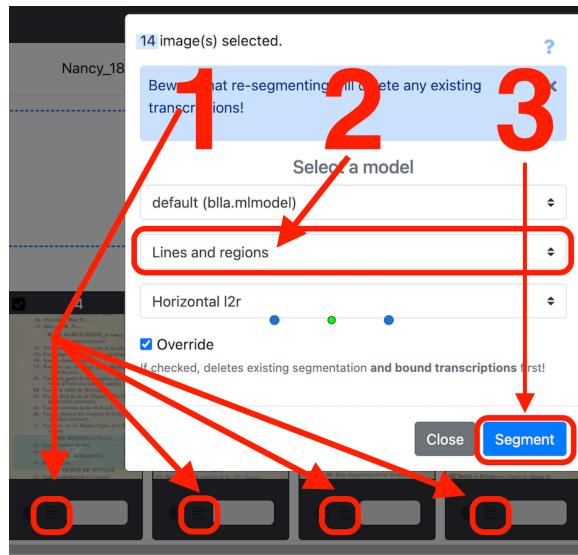


FIGURE 5.15 – eScriptorium : reconnaissance des lignes

L'étape suivante consiste à lancer une transcription automatique du texte contenu dans les images. Comme indiqué sur la figure 5.16, l'utilisateur doit [1] cliquer sur n'importe quel bouton "transcrire" d'une des images cochées, ou bien sur le bouton général **Transcribe**. Il faut après [2] sélectionner le modèle "19th Century prints - HTRcatalogs Artlas" (ou bien celui du choix de l'utilisateur). Finalement, il faut [3] cliquez sur **Transcribe**. Une fois une image transcrit, son bouton "transcrire" indique la complétion à 100%.

L'utilisateur peut à présent consulter et éditer les transcriptions en cliquant sur le bouton "éditer" des icônes ; il le mènera vers la page d'édition de chaque image, où il

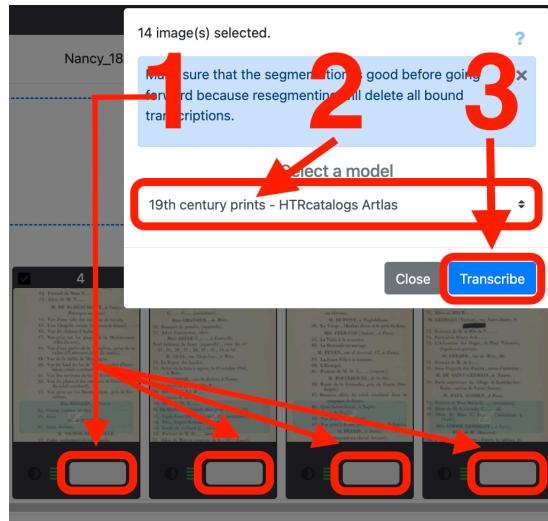


FIGURE 5.16 – eScriptorium : lancer la transcription

pourra corriger manuellement la transcription automatique, puis segmenter manuellement les régions.

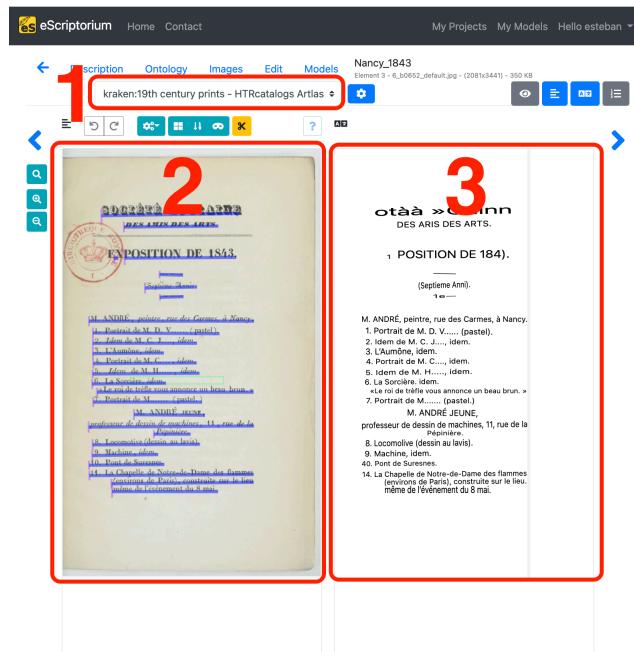


FIGURE 5.17 – eScriptorium : page d'édition

Comme indiqué sur la figure 5.17, pour que la transcription s'affiche correctement, il faut [1] choisir la transcription qui correspond au modèle de reconnaissance de caractères choisi. Sur le viseur de la transcription, on trouvera à gauche [2] l'image courante, avec la segmentation des régions et des lignes (si ces segmentations existent), et à droite [3] la transcription du texte contenu dans l'image (si elle existe).

Pour corriger la transcription (voir figure 5.18), l'utilisateur doit [1] cliquer sur la ligne qu'il souhaite corriger dans la section "transcription" (à droite), puis [2] écrire le texte souhaité sur l'éditeur qui apparaît. Le catalogue Nancy 1843 fournit un bon exemple

d'une transcription non satisfaisante, puisque le texte "SOCIÉTÉ LORRAINE" a été très mal reconnu. Cela est normal, puisqu'il ne s'agit pas de caractères typiques :

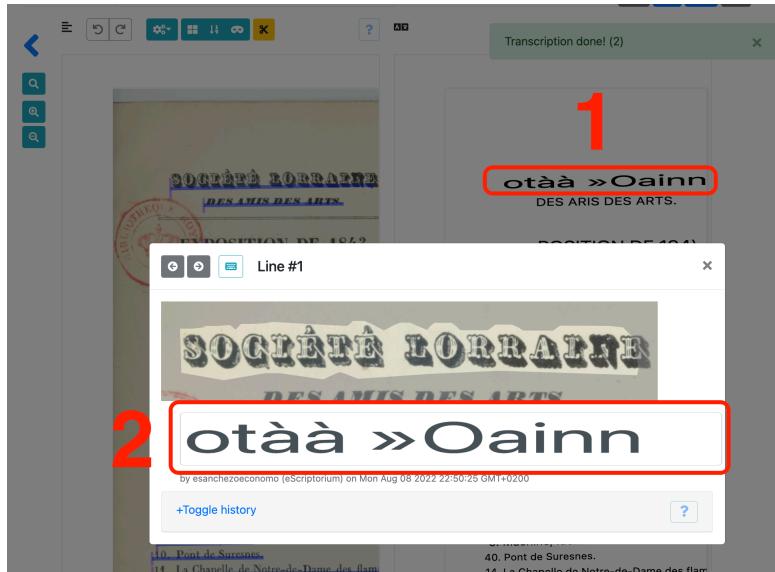


FIGURE 5.18 – eScriptorium : corrections manuelles

Il suffit alors de substituer la correction au texte affiché, puis de cliquer sur ENTER ou sur le bouton de flèche vers le bas afin d'aller vers la ligne suivante. L'utilisateur doit vérifier chaque ligne de chaque page, et faire toutes les corrections nécessaires. Il est important que les ponctuations soient correctement transcrits, et en particulier les caractères qui signalent des œuvres (par exemple "1. titre de l'œuvre"). Il faut vérifier les numéros, mais aussi les points et les tirets qui délimitent le numéro et le titre de l'œuvre, les virgules qui séparent les numéros des adresses, etc. ; ces éléments sont importants pour que le script soit capable de reconnaître des sections différentes pour reconstituer les entrées du catalogue. L'utilisateur n'a pas besoin de sauvegarder ses corrections, cela se fait automatiquement au fur et à mesure des modifications.

5.4 Segmentation manuelle des images

Une fois la correction de la transcription automatique terminée, l'utilisateur peut faire la segmentation manuelle des images. Cela se passe dans la section de gauche, qui contient l'image de la page. Cette section sert à signaler sur la page des régions et des lignes détectées automatiquement ou saisies manuellement.

Par défaut, en cliquant sur l'image, l'utilisateur dessine des lignes. Il peut (et parfois doit) modifier des lignes qui sont mal saisies ou détectées. La figure 5.19 inclut les cas les plus communs : elles peuvent [1] ne pas couvrir tout le texte, [2] déborder des régions, ou [3] couvrir des espaces où il n'y a pas de texte. Il est possible de les manipuler en cliquant dessus, ou des les éliminer en cliquant dessus puis sur le bouton rouge "poubelle" à gauche

de l'image.

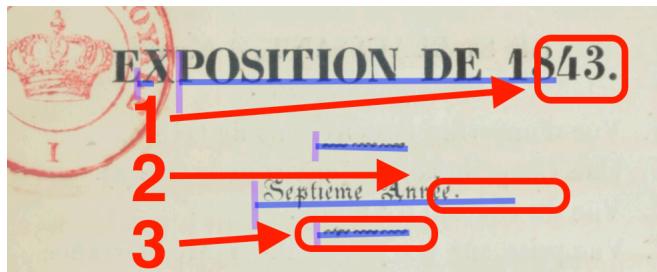


FIGURE 5.19 – eScriptorium : lignes avec erreurs

La figure 5.20 montre comment dessiner des régions. L'utilisateur doit [1] cliquer sur le bouton "Switch to region mode", puis [2] saisir la région sur l'image avec la souris. Il faut aussi indiquer le nom de la zone : il faut [3] cliquer sur le bouton "T", puis [4] sélectionner la zone concernée et appuyez sur ENTER. C'est ici que l'ontologie Segmonto, saisie précédemment, est affichée et disponible pour le nommage des zones.

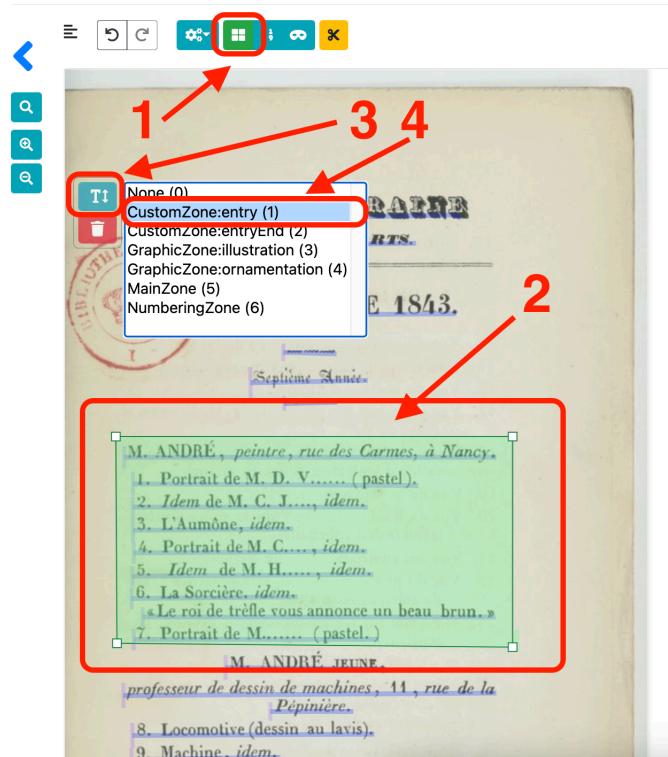


FIGURE 5.20 – eScriptorium : saisir des zones

Normalement, l'ordre de saisie des régions aura un impact sur l'imbrication des éléments dans le fichier ALTO en output. Cela posait problème et impliquait que tout document saisi dans un ordre différent à celui compris par le script ne puisse être traité : avant notre mission, c'était le cas de la plupart des catalogues dans le corpus de transcriptions. Le script est actuellement capable de restructurer ces imbrications et de comprendre tous les documents ALTO en entrée, raison pour laquelle l'ordre de saisie des régions ne

posera aucun problème. Il est tout de même important de faire noter à l'utilisateur cette situation, afin qu'il prenne conscience sur les bonnes pratiques de saisie des régions sur eScriptorium. Il faut commencer par les plus proches du texte, puis les plus amples et générales. Une fois les `CustomZone:entry`, `CustomZone:entryEnd` et `NumberingZone` saisies (ou tout autre type de zone), il faudra saisir la `Mainzone`, qui couvre tout l'espace sémantique de la page (donc pas le numéro de page).

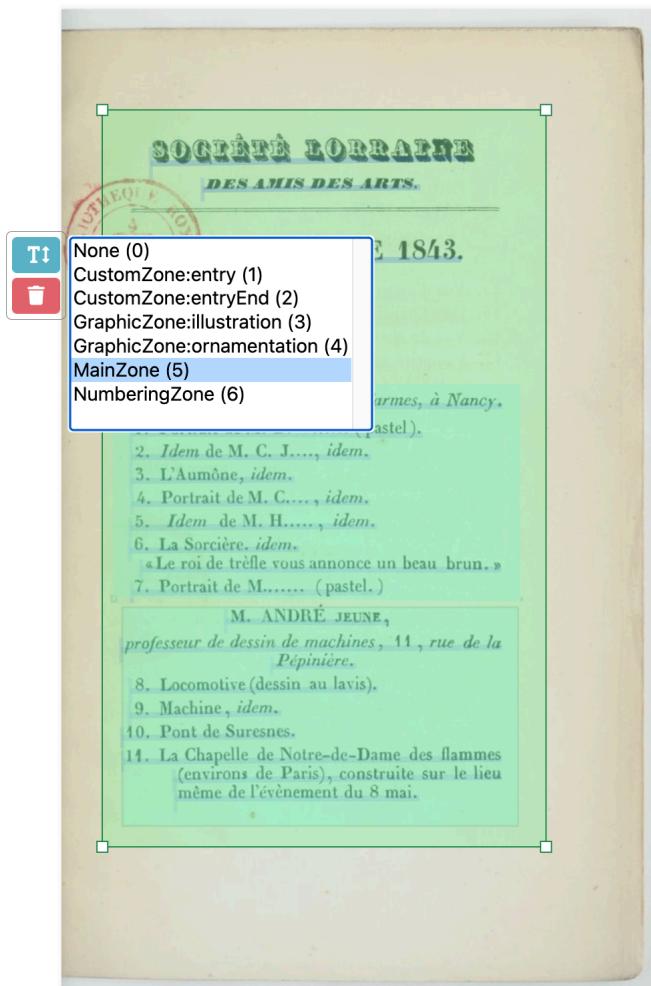


FIGURE 5.21 – eScriptorium : MainZone

L'utilisateur doit segmenter manuellement toutes les pages du catalogue qui contiennent des entrées. Il devra notamment veiller à ce qu'il n'y ait pas des conflits d'imbrication (des lignes qui dépassent leurs régions, des zones `CustomZone:Entry` qui dépassent la `MainZone`, etc.). Dans le cadre du catalogue Nancy 1843, nous allons segmenter une page sans entrées et cinq pages avec des entrées (pages 3-7) ; cela permettra de montrer comment le script traite une page qui ne contient pas d'entrées.

5.5 Output ALTO

À présent, l'utilisateur a transcrit et segmenté ses pages. Il faut maintenant récupérer le fichier ALTO en **output** d'eScriptorium afin de l'utilser comme **input** du script `extractionCatalogs`.

Pour récupérer ces fichiers, comme indiqué sur la figure 5.22, l'utilisateur doit [1] cliquer sur l'onglet "Images" pour retourner sur la frise des images, puis [2] cocher sur les images à télécharger et [3] cliquer sur "Export". Il doit veiller à respecter les indications de la figure 5.23 : [1] le modèle de reconnaissance de caractères de l'export doit correspondre à celui qui a été utilisé, [2] les documents en **output** doivent être des fichiers ALTO, il doit [3] cocher sur "include images" et [4] vérifier que toutes les régions de l'ontologie soient cochées. [5] En cliquant sur "Export", un avertissement vert apparaît en haut à droite de la page si celui-ci est réussi, dans lequel il faut cliquer sur "Download" (figure 5.24).

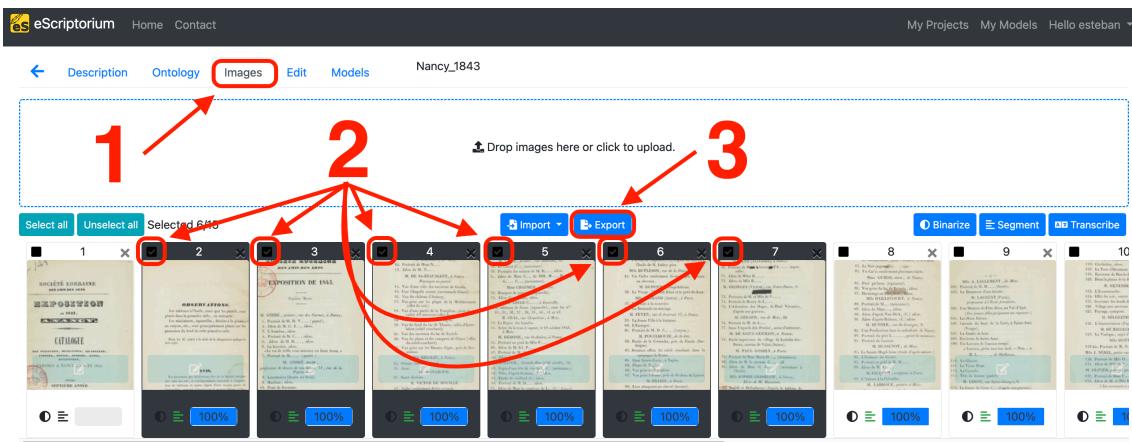


FIGURE 5.22 – eScriptorium : télécharger les fichiers ALTO

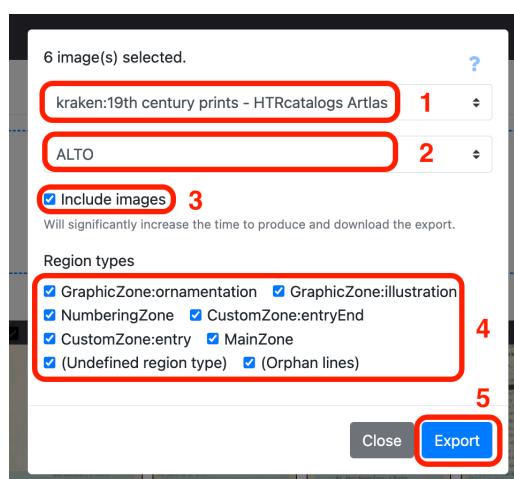


FIGURE 5.23 – eScriptorium : modalités d'export

L'utilisateur télécharge ainsi un fichier ZIP qui contient les images du catalogue, et autant de fichiers ALTO4 contenant leurs transcriptions et leurs segmentations. Il peut



FIGURE 5.24 – eScriptorium : Download

être utile de renommer les fichiers afin de les rendre plus compréhensibles à l’oeil humain, mais cela n’est pas obligatoire et implique de manipuler les fichiers ALTO. Cela n’est pas obligatoire, tant que les fichiers gardent un ordre numérique, mais il peut-être intéressant pour les étudiants de s’aventurer à faire des changements manuels sur ce type de fichier. Pour le catalogue Nancy 1843, on va suivre la convention de nommage suivante :

Cat_Nancy_1843_typo_0000.jpg – Cat_Nancy_1843_typo_0005.jpg
 Cat_Nancy_1843_typo_0000.xml – Cat_Nancy_1843_typo_0005.xml

Après avoir renommé chaque fichier, il faut actualiser le nom de l’image à laquelle chaque fichier ALTO fait référence, afin que les deux fichiers restent liés. Notons pourtant qu’il s’agit uniquement d’exposer des bonnes pratiques, puisque cela ne va affecter en rien le programme extractionCatalogs. Pour le faire, il faut ouvrir chaque fichier ALTO et actualiser le nom de l’image (.jpg) à laquelle ce fichier fait référence dans la balise <filename> (voir la figure 5.25)

```
<?xml version="1.0" encoding="UTF-8"?>
<alto xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xmlns="http://www.loc.gov/standards/alto/ns-v4#"
      xsi:schemaLocation="http://www.loc.gov/standards/alto/ns-v4# http://www.loc.gov/standards/alto/v4/alto-4-2.xsd">
  <Description>
    <MeasurementUnit>pixel</MeasurementUnit>
    <sourceImageInformation>
      <fileName>Cat_Nancy_1843_typo_0000.jpg</fileName>
      <fileIdentifier>https://gallica.bnf.fr/ark:/12148/bpt6k62251805/f6/full/full/0/default.jpg</fileIdentifier>
    </sourceImageInformation>
  </Description>
```

FIGURE 5.25 – ALTO : image liée

ALTO⁶ (Analysed Layout and Text Object) est un standard XML permettant de rendre compte de la mise en page physique et de la structure logique d’un texte transcrit par reconnaissance optique de caractères (OCR). Si la transcription est correcte et que les zones de la page sont correctement saisies sur eScriptorium, le script extractionCatalogs sera extrêmement performant au moment de l’extraction et de la reconstruction des entrées du catalogue.

La structure fondamentale d’un fichier ALTO est constituée par une balise racine <alto> contenant des sous-ensembles <Description>, <Tags> et <Layout> :

- <Description> : métadonnées sur le fichier source. Le nom de l’image est indiqué, mais aussi le lien vers l’image iiif quand le fichier est issu d’un manifeste .

6. ALTO Editorial Board (éd.), *ALTO : Technical Metadata for Layout and Text Objects*, 2004, URL : <https://www.loc.gov/standards/alto/> (visité le 09/01/2022).

```

<Description>
  <MeasurementUnit>pixel</MeasurementUnit>
  <sourceImageInformation>
    <fileName>Cat_Nancy_1843_typo_0005.jpg</fileName>
    <fileIdentifier>https://gallica.bnf.fr/iiif/ark:/12148/bpt6k62251805/f11/full/full/0/default.jpg</fileIdentifier>
  </sourceImageInformation>
</Description>

```

FIGURE 5.26 – ALTO : Description

- <Tags> : établit des ID pour signaler les noms des régions (des tags). C'est ici que nous pourrons retrouver l'ontologie Segmonto. Ces ID seront utilisés dans la balise <Layout> pour signaler à quoi correspond chaque bloc de texte.

```

<Tags>
  <OtherTag ID="BT6" LABEL="GraphicZone:illustration"
    DESCRIPTION="block type GraphicZone:illustration"/>
  <OtherTag ID="BT2" LABEL="MainZone" DESCRIPTION="block type MainZone"/>
  <OtherTag ID="BT1" LABEL="TitlePageZone" DESCRIPTION="block type TitlePageZone"/>
  <OtherTag ID="BT8" LABEL="NumberingZone" DESCRIPTION="block type NumberingZone"/>
  <OtherTag ID="BT145" LABEL="RunningTitleZone" DESCRIPTION="block type RunningTitleZone"/>
  <OtherTag ID="BT15" LABEL="CustomZone:entry" DESCRIPTION="block type CustomZone:entry"/>
  <OtherTag ID="BT16" LABEL="CustomZone:entryEnd" DESCRIPTION="block type CustomZone:entryEnd"/>
  <OtherTag ID="BT13" LABEL="StampZone" DESCRIPTION="block type StampZone"/>
  <OtherTag ID="LT1" LABEL="DefaultLine" DESCRIPTION="line type DefaultLine"/>
</Tags>

```

FIGURE 5.27 – ALTO : Tags

- <Layout> : Contient tout le texte transcrit. Nous retrouverons dedans des balises <TextBlock> : chacune correspond à une des régions signalées manuellement sur eScriptorium. Leur Attribut TAGREFS fait référence à l'ID qui permet de connaître le nom de la zone dans la balise <Tags>. La balise <Shape> indique la position du TextBlock dans la page. Un TextBlock doit contenir des balises <TextLine>, qui elles mêmes contiennent des balises <Shape> et des balises <String>. C'est dans l'attribut CONTENT de cette dernière que se trouve le texte transcrit. Dans l'exemple de la figure 5.28, l'attribut TAGREFS="BT15" fait référence au type de région CustomZone:entry.

```

<TextBlock HPOS="524" VPOS="489" WIDTH="1110" HEIGHT="262" ID="eSc_textblock_b137d3fb"
  TAGREFS="BT15">
  <Shape>
    <Polygon POINTS="533 489 524 751 1615 751 1634 501"/>
  </Shape>
  <Textline ID="tL_1" TAGREFS="LT1" BASELINE="864 542 933 544 1014 540 1373 544 1556 549"
    HPOS="864" VPOS="497" WIDTH="692" HEIGHT="66">
    <Shape>
      <Polygon POINTS="864 497 1556 497 1556 563 864 563"/>
    </Shape>
    <String CONTENT="M. BASTIEN, de Metz." HPOS="864" VPOS="497" WIDTH="692" HEIGHT="66"/>
  </Textline>
  <Textline ID="tL_2" TAGREFS="LT1"
    BASELINE="552 625 628 627 876 625 958 626 1109 626 1277 628" HPOS="552" VPOS="577"
    WIDTH="726" HEIGHT="51">
    <Shape>
      <Polygon POINTS="552 577 1278 577 1278 628 552 628"/>
    </Shape>
    <String CONTENT="12. Portrait de Mme N...." HPOS="552" VPOS="577" WIDTH="726"
      HEIGHT="51"/>
  </Textline>
</TextBlock>

```

FIGURE 5.28 – ALTO : Layout

L'utilisateur peut corriger directement ces fichiers ALTO (pour préciser le découpage et les imbrications des régions, le contenu des lignes, les noms des zones), mais cela risque d'être très difficile pour un néophyte. Normalement, toutes les corrections (transcriptions et segmentations) sont faisables sur l'interface eScriptorium, ou bien sur le tableau produit par le programme extractionCatalogs (il est donc conseillé aux étudiants de corriger directement l'`output`). Cependant, il est possible et parfois souhaitable, dans un cadre pédagogique, de corriger les fichiers XML (ALTO, TEI) pour observer les conséquences que cela a sur le fonctionnement du script extractionCatalogs.

Cette possibilité d'approfondissement des difficultés peut être intéressante pour aborder des groupes d'étudiants avec des décalages plus ou moins importants de culture numérique. Si l'utilisation de l'interface ou du guide sur le Notebook leur est trop facile, ils peuvent explorer les logiciels et technologies que ce dernier leur présente : iiif, fichiers ALTO, script python, fichier TEI produit (puisque celui-ci est utilisé pour produire le tableau CSV).

Chapitre 6

Utiliser le script python ”extractionCatalogs”

Une fois le travail de transcription automatique, de segmentation manuelle et de correction terminé, l'utilisateur téléchargera sur la plateforme eScriptorium un dossier contenant les images de son catalogue (elle seront toujours en format .jpg) et autant de fichiers XML ALTO contenant l'encodage du texte et des zones.

Le dossier peut être placé n'importe où en local ; il faudra indiquer le chemin au script extractionCatalogs. Ce chemin peut-être absolu (partir de la racine de l'ordinateur), ou bien relatif au dossier du script. Il peut-être est donc utile de créer un dossier à l'intérieur (ou proche) du dossier `extractionCatalogs`. Pour le catalogue Nancy 1843, nous avons créé un dossier `exemples_guide` dans `extractionCatalogs`. Ce dossier contient d'autres exemples qui sont évoqués aussi bien dans ce chapitre que dans le guide d'utilisation simplifié Jupyter Notebook.

Le script extractionCatalogs va chercher et traiter les fichiers ALTO contenus dans le dossier indiqué. Il va utiliser les informations de segmentation pour pour reconstituer les entrées, ainsi que des expressions régulières pour analyser chaque ligne et déterminer si elle correspond à un exposant, à une œuvre ou à des informations complémentaires. Dans certains cas, il sera possible de déterminer la nature des informations complémentaires.

Le script est l'objet d'améliorations continues. En son état actuel, il est performant aussi bien pour ses visées pédagogiques que pour la production de données satisfaisantes. Il peut tout de même faire l'objet d'améliorations et d'évolutions, raison pour laquelle il est aussi important de le consulter directement pour comprendre en profondeur son fonctionnement. Dans le cadre du projet pédagogique du certificat en humanités numériques, mais aussi en prévision des stages à venir, une partie conséquente du développement du prototype établi par Juliette Janès en 2021 à consisté à améliorer son arborescence, à améliorer sémantiquement les noms des fonctions et des variables, à consolider les descriptions des fonctions, et à commenter profusément le code pour le rendre rapidement compréhensible. L'utilisateur peut donc ouvrir les fichier .py du dossier `extractionCatalogs`, qui

contiennent tout le programme python. Il n'aura pas besoin d'intervenir dessus, et sera accompagné tout au long de sa lecture.

6.1 Commandes du script

La commande pour lancer le script a été simplifiée à l'extrême. Avant notre mission, il fallait faire une analyse préalable du catalogue pour indiquer au script, d'après une typologie, quelle était sa structure. Il n'était pas possible de choisir l'emplacement, et il fallait indiquer deux fois le nom du catalogue (une en tant qu'ID, un deuxième en tant que nom du fichier TEI). De plus, il y avait de nombreuses contraintes/erreurs techniques qui rendaient le lancement problématique. En l'état actuel, le script marche avec la totalité des catalogues du Corpus (37), contre 6 précédemment. En plus, le script est capable d'indiquer sur le terminal de manière claire pour un néophyte s'il y a des erreurs ou des problèmes qui l'empêchent de fonctionner.

Voici le modèle de commande pour lancer le script :

```
!python3 run.py input output titre
```

Pour traiter le catalogue Nancy 1843, nous devons construire la commande suivante :

```
!python3 run.py exemples_guide/Cat_Nancy_1843_page/ exemples_guide/extractions/ Nancy_1843
```

Voici ses composantes :

- ! indique à Jupyter Notebook que nous voulons lancer la commande sur le terminal. Si l'utilisateur lance la commande directement sur un terminal, il ne faut pas utiliser ce point d'exclamation
- python3 indique que le script doit être exécuté en langage python 3
- run.py est le fichier python que nous voulons exécuter. Tous les autres fichiers python se trouvent dans des sous-dossiers et sont appelés par celui-ci
- exemples_guide/Cat_Nancy_1843_page/ est le chemin vers les fichiers ALTO produits par eScriptorium. Ce dossier contient uniquement une page de catalogue (fichier .jpg + ALTO) afin que les premiers exemples restent simples
- exemples_guide/extractions/ est le chemin pour le dossier d'extraction que le script va produire. Si le chemin ou le dossier n'existent pas, le script les crée
- Nancy_1843 est le nom souhaité pour le catalogue. Il sera utilisé pour le noms des fichiers en **output** (TEI et CSV) ainsi que pour les identifiants constitutifs des fichiers XML produits (TEI, restructurations ALTO)

Le guide Jupyter Notebook permet de lancer et de visualiser les résultats de cette commande en exécutant la cellule avec le code correspondant (COMMAND/ENTER, SHIFT/ENTER ou bouton "Run"). La cellule produit le message de terminal que l'on peut

```
In [1]: !python3 run.py exemples_guide/Cat_Nancy_1843_page/ exemples_guide/extractions/ Nancy_1843_page
```

```
=====
1 - Traitement de Cat_Nancy_1843_typo_0001.xml
  1 - Vérification de la conformité à l'ontologie Segmonto
    ✓ le fichier issu d'eScriptorium est conforme à l'ontologie Segmonto
  2 - Restructuration du fichier
    ✓ fichier 'Cat_Nancy_1843_typo_0001.xml_restructuration.xml' créé
  3 - Vérification de l'imbrication des zones saisies sur eScriptorium
    Le Fichier Alto produit par eScriptorium contient des zones ['TextBlock'] vides
    - Parmi ces zones ['TextBlock'] vides, il y a des 'CustomZone:entry'
    ✓ Correction automatique : 18 lignes correspondant à 2 zones vides ont été déplacées vers ces zones
  4 - Extraction :
```

```
M.ANDRÉ,
peintre, rue des Carmes, à Nancy.
  1. Portrait de M.D.V.....  

(pastel).
  2. Idem de M. C. J....., idem.  

idem.
  3. L'Aumône, idem.  

idem.
  4. Portrait de M. C...., idem.  

idem.
  5. Idem de M. H....., idem.  

idem.
  6. La Sorcière. idem.
  7. Portrait de M.....  

(pastel.)
Image : https://gallica.bnf.fr/iiif/ark:/12148/bpt6k62251805/f7/217.0,1320.0,1333.0,735.0/full/0/de  

fault.jpg
M. ANDRÉ JEUNE,  

professeur de dessin de machines, 11, rue de la Pépinière.
  8. Locomotive  

(dessin au lavis).
  9. Machine, idem.  

idem.
  10. Pont de Suresnes.
  11. La Chapelle de Notre-de-Dame des flammes même de l'évènement du 8 mai.
Image : https://gallica.bnf.fr/iiif/ark:/12148/bpt6k62251805/f7/203.0,2079.0,1354.0,714.0/full/0/de  

fault.jpg
```

```
=====
Résumé :
  1 page de catalogue traitée [fichier ALTO]
  2 entrées de catalogue extraites [TextBlocks 'CustomZone:entry']
  2 exposants signalés
  11 œuvres extraites

Analyse des fichiers en input [ALTO] :
  ✓ La segmentation des pages est conforme à l'ontologie Segmonto
Analyse du fichier en output [TEI] :
  ✓ Le document XML produit est conforme au schéma TEI et à l'ODD du projet.
Création d'un tableau csv :
  ✓ Le fichier csv a été produit

Chemin du dossier d'extraction : /Users/EstebanSanchez/TNAH_Git/IMAGO-Artlets/Github_repos/Juliette_Janès/IMAGO-Catalogues-Jjanès/extractionCatalogs/exemples_guide/extractions/extraction_Nancy_1843_page
```

FIGURE 6.1 – Script : premier exemple d'extraction

observer sur la figure 6.1 : pour chaque page traitée, l'utilisateur obtient un récapitulatif concis et clair des étapes, ainsi qu'une visualisation structurée de l'extraction. Une fois toutes les pages traitées (l'exemple n'en contient qu'une), l'utilisateur obtient un résumé avec des statistiques générales du traitement de l'ensemble des pages. L'annexe C contient une versions en texte brut de cet output de terminal.

La dernière information signale le chemin dans lequel l'extraction a été déposée. Dans le cas de l'exemple, il s'agit du dossier `extraction_Nancy_1843_page`. L'utilisateur peut consulter dedans les fichiers suivants :

- `Nancy_1843_page.xml` : un fichier XML contenant la totalité des entrées extraites
- CSV : un dossier CSV avec deux tableurs :

 - `Nancy_1843_page_tableau_simple.csv` : tableur simplifié pour lecture humaine
 - `Nancy_1843_page_tableau.csv` : tableur complet et utilisable comme input

pour la base de données Artl@s

- **restructuration ALTO** : un dossier avec des fichiers ALTO ”restructurés” ”re-segmentés” : ils résultent d’une correction automatique des erreurs de saisie sur eScriptorium
- éventuellement, un fichier `.txt` indiquant des problèmes d’extraction (dans le cas de l’exemple, aucune erreur n’a été détectée)

6.2 Fonctionnement du script

Le guide Jupyter Notebook accompagne l’utilisateur dans le développement d’une culture numérique suffisante pour comprendre dans son ensemble l’**output** du script (messages de terminal, fichiers produits). Il lui permet aussi de comprendre la structure fondamentale du dossier contenant le script, ainsi que le code contenu. En fonction de ses besoins, il pourra consulter les fichiers pour accéder à des détails beaucoup plus approfondis, les script étant accompagné d’explications profuses et simples.

En tout état de cause, le Notebook, le script et de manière plus générale l’ensemble du pipeline ont été conçus pour qu’un utilisateur sans connaissances informatiques particulières puisse les utiliser, sans intervenir dans leur mécanique interne. Il suffit qu’il développe les capacités nécessaires (pour cela, il est accompagné par le Notebook et par ses enseignants) pour l’exécuter et pour pouvoir comprendre les messages et fichiers en **output**.

Cette section décrit le fonctionnement du script à travers des étapes fondamentales exécutées par le fichier `run.py`. Le dossier `extractionCatalogs` contient :

- `run.py` : fichier python principal qui exécute tous les fichiers `.py` secondaires, contenus dans les sous-dossiers `extractionCatalogs/extractionCatalogs/fonctions` et `extractionCatalogs/extractionCatalogs/variables`
- `README.md` : fichier expliquant comment installer le script (ce sont les mêmes instructions que dans la partie 0. **Introduction** du guide Notebook. Ces instructions sont expliquées de manière plus approfondie dans le chapitre 7 du mémoire)
- `requirements.txt` : fichier `.txt` indiquant les paquets python externes dont le script à besoin pour fonctionner. Ces paquets sont installés lors de l’étape d’installation avec la commande `!pip install -r requirements.txt`)
- dossiers `exemples` : dossiers contenant des catalogues à l’état d’encodage ALTO ou d’extraction python :
 - `exemples_guide` : exemple utilisés dans le guide Jupyter Notebook
 - `exemples_input` : totalité des catalogues du corpus Artl@s encodés en fichiers ALTO (sortie eScriptorium). Ils sont issus du travail collaboratif des stagiaires, étudiants et chercheurs contribuant au projet Artl@s. Le dossier contient pour

l'instant des encodages relatifs à 37 catalogues d'exposition d'art du XIXe et XXe siècles.

- `exemples_output` : extractions python pour chaque catalogue du corpus avec le script développé pendant le stage 2022. Ces dossiers permettent de constater les atouts et les limites actuelles du script qui fonctionne globalement de manière très satisfaisante. Quand il y a des soucis d'extraction, le script est systématiquement capable d'expliquer les problèmes.
- `Guide - extraction de catalogues.ipynb` : guide Jupyter Notebook
- dossier `env` : environnement virtuel contenant les paquets python. Ce dossier n'existe que dans les cas où le script a été téléchargé hors de Google Colab (normalement, cela aura lieu sur Github)

`run.py` est le fichier exécuté par la commande `!python3 run.py`, suivie des indications secondaires (`input`, `output`, nom du catalogue). Ce fichier est la colonne vertébrale du script, puisqu'il contient tout le processus du début à la fin. Il appelle les autres fichiers `.py`, qui fonctionnent comme des extensions séparées pour que le script garde une structure logique et compréhensible par un être humain sous la forme d'une arborescence de dossiers. Ces autres fichiers `.py` sont des fonctions spécialisées, souvent longues, qui ont été rangées dans des sous-dossiers. Le fichier `run.py` exécute les opérations indiquées dans la liste qui suit. La même numérotation est utilisée dans les commentaires du fichier : il est donc possible de suivre la liste suivante et de parcourir le script en même temps.

1.1 On appelle les paquets externes, modules et fonctions nécessaires : nous allons faire appel, dossier par dossier et fichier par fichier, aux fonctions créées pour ce script et situées en dehors de `run.py` :

```
from extractionCatalogs.fonctions.extractionCatEntrees import extInfo_Cat
from extractionCatalogs.fonctions.creationTEI import creation_header
from extractionCatalogs.fonctions.restructuration import restructuration_automatique #...
from extractionCatalogs.fonctions.Validations_xml import conformite_Segmento, association_xml_rng
from extractionCatalogs.fonctions.XMLtoCSV import XML_to_CSV, csv_immediat
from extractionCatalogs.fonctions.automatisation_kraken.kraken_automatic import transcription
from extractionCatalogs.variables import contenu_TEI
from extractionCatalogs.fonctions.extractionCatEntrees_fonctions import ordonner_altos #...
```

Ces modules sont extraits des fichiers suivants :

- `extractionCatEntrees.py`
- `creationTEI.py`
- `restructuration.py`
- `Validations_xml.py`
- `XMLtoCSV.py`
- `variables.py`
- `extractionCatEntrees_fonctions`

Nous allons aussi appeler des paquets externes au script. Certains d'entre eux ont été installés avec le fichier `requirements.txt`, d'autres sont intégrés à python3 et

peuvent être appelés sans installation préalable. Pour notre script, nous avons besoin des paquets python suivants :

- `lxml` : implémentation xml pour python, qui nous permettra de créer et de manipuler de objets xml
- `click` : permet de créer les commandes pour exécuter le script (`input`, `output`, `titre`)
- `re` : implémentation regex pour python, qui nous permettra de construire et utiliser des expressions régulières
- `os` : permet de manipuler avec python des fichiers et dossiers présents sur l'ordinateur

- 1.2 Création des commandes pour lancer le script sur le Terminal : le paquet `click` permet d'établir une communication avec le terminal, afin de demander en `input` des variables spécifiques. Dans notre cas, outre les commandes fondamentales `!python3 run.py`, nous utilisons `click` pour créer et demander d'autres éléments en `input` : `directory`, `output` et `titlecat`. Ce sont des chaînes de caractère que le script va parser pour en vérifier la validité et pour les utiliser en tant que variables.
- 1.3 Création de la fonction principale du script : nous définissons la fonction `extraction()` qui constitue la totalité du script, puisqu'elle appelle toutes les fonctions secondaires et qu'elle produit les documents en sortie (TEI, CSV, restructurations ALTO et fichier `problèmes.txt`) :

```
def extraction(directory, output, titlecat, segmentationtranscription)
    # contenu de la fonction (+500 lignes)
```

Tout ce qui suit est contenu dans cette fonction.

- 2.1 Création d'un dossier pour les `output` : nous utilisons le paquet `os` pour traiter les chemins indiqués (`input` et `output`), et pour créer les chemins qui seront nécessaires au script pour aller chercher puis produire les fichiers.
- 2.2 Création immédiate d'un fichier CSV, si la chaîne "csv" est contenue dans le titre du catalogue (variable `titlecat` passée avec la commande) : lorsque le titre du catalogue indiqué dans la commande se termine par `.csv`, le script va tout simplement produire un tableau avec les fichiers qui existent déjà, sans traiter les fichiers ALTO en `input` ni créer de fichier TEI. Cela est utile lorsqu'on a déjà exécuté préalablement le script, que nous avons fait des modifications manuelles sur le fichier TEI, et que nous voulons simplement actualiser notre tableau CSV. Dans ce cas, le script s'arrêtera immédiatement à cette étape.
- 2.3 création du fichier `problèmes.txt` et traitement de l'option `-st` : on vérifie les chemins existants pour créer (ou remplacer) le fichier `problèmes.txt`. Nous traitons aussi une commande en option, `-st`, qui permet de demander une transcription

automatique au logiciel kraken. Dans ce cas, l'input devra être constitué uniquement d'images. Cette option, développée lors du stage 2021, ne fonctionne pas pour plusieurs raisons : les commandes kraken indiquées ne sont plus d'actualité, la segmentation automatique donne des résultats très défaillants, et la transcription automatique ne peut pas être corrigée. L'option `-st` reste donc une implémentation expérimentale, en prospective d'avancements techniques hypothétiques qui permettraient de faire des transcriptions/segmentations automatiques suffisamment performantes comme pour ne plus avoir à utiliser eScriptorium. Dans ce cas, le script fonctionnerait de manière autonome et constituerait la totalité du pipeline. Il serait alors intéressant de pouvoir passer en commande directement un lien iiif comme ensemble d'images à traiter.

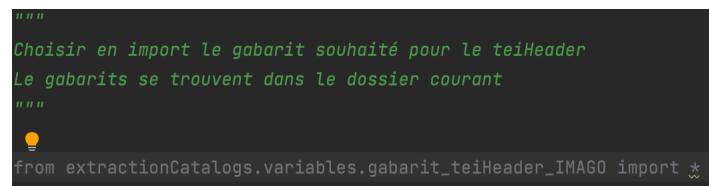
- 3 Création d'un arbre TEI : nous utilisons le paquet `lxml` pour construire un arbre TEI conforme au projet (lequel qui possède un schéma et une ODD spécifiques développés lors du stage 2020)¹. Le script crée, un par un, les sous-éléments des l'arbre (`tei`, `teiHeader`, `text`, `body`, `list`, etc.). Pour créer le `teiHeader`, le script utilise le gabarit suivant :

```
variables/gabarit_teihandler_IMAGO.py
```

Ce document est adapté au projet Artl@s et permet que les métadonnées de l'en-codage soient complètes et facilement modifiables à travers de variables. Cela facilite les modifications, puisqu'il n'est plus nécessaire d'intervenir sur la fonction de construction de l'arborescence TEI. Il est possible de construire d'autres gabarits adaptés à d'autres projets, grâce au document suivant :

```
variables/gabarit\_\teihandler\_\vide.py
```

Il faut remplir dedans les noms des variables selon les besoins du projet, et en indiquant le nom de ce document sur le fichier `variables/contenu_teihandler.py` :



```
"""
Choisir en import le gabarit souhaité pour le teihandler
Le gabarits se trouvent dans le dossier courant
"""

from extractionCatalogs.variables.gabarit_teihandler_IMAGO import *
```

FIGURE 6.2 – Script : gabarit personnalisé pour le teihandler

Cette fonctionnalité à déjà été utilisée par une étudiante du certificat en humanités numériques de l'Unige, Frédérique Pradier, dont on a intégré au dossier le gabarit :

```
variables/gabarit\_\teihandler\_\PictoCatalogs.py
```

1. C. Corbières, *Du Catalogue Au Fichier TEI : Crédation d'un Workflow Pour Encoder Automati-quement En XML-TEI Des Catalogues d'exposition...*

4.1 Traitement préalable des ALTO en input : nous allons maintenant aller chercher les fichiers ALTO contenus dans le dossier signalé par la commande. Nous allons faire une boucle afin que chaque fichier suive le même traitement, un par un.

```
for fichier in liste_en_ordre:
    # contenu de la fonction (+100 lignes)
```

- 4.2 On analyse la conformité et la structure des fichiers ALTO : une fonction secondaire permet de vérifier si les zones nommées correspondent à l'ontologie Segmonto. Les erreurs éventuelles seront signalées dans le fichier **problèmes.txt**
- 4.3 Restructuration des ALTO en input : une première restructuration a lieu. Parfois, les fichiers ALTO produits par eScriptorium présentent les éléments **TextBlock** (qui correspondent à des entrées de catalogue) de manière désordonnée. Il s'agit de les mettre dans le bon ordre avec la feuille de transformation XSL **Restructuration_alto.xsl**. Un dossier **restructuration ALTO** est créé et les fichiers restructurés y sont déposés.
- 4.4 Restructuration eventuelle de la segmentation des ALTO en input ("resegmentation" est le mot utilisé dans le nom des fichiers pour différencier cette deuxième restructuration) : il existe un autre problème, beaucoup plus complexe, dans les fichiers ALTO produits par eScriptorium. Selon l'ordre de saisie des lignes et des régions sur l'interface, le fichier ALTO produit peut avoir des structures très variables : des fois, les **TextBlock** qui contiennent le texte transcrit ne correspondent pas à des **Custom:entry** (comme attendu), mais à des **MainZone**, ce qui rend le fichier intraitable par le script. D'autres erreurs dans ce registre peuvent avoir lieu. Nous avons donc créé une fonction qui utilise les informations de positionnement spatial de chaque **TextBlock** pour corriger toutes les erreurs d'imbrication du fichier. Si des erreurs sont détectées, la fonction les corrige dans des fichiers "resegmentation" qui vont être déposés à côté des fichiers "restructuration". Cette fonction à l'intérêt de corriger un problème inhérent à l'interface eScriptorium en calculant les régions selon les informations du fichier ALTO. Elle peut donc être reprise dans le cas d'autres projets pour adapter les imbrications de l'output selon les besoins. Cette fonction a permis de passer de 6 catalogues traitables à 37 dans le corpus Artl@s d'encodages en ALTO.

```
for TextBlock in TextBlocks_vides:
    # on récupère l'ID du TextBlock courant, et on l'utilise pour construire le chemin xpath vers les
    # attributs qui signalent sa position dans la page :
    TextBlock_ID = TextBlock.xpath("@ID", namespaces=NS)[0]
    # on utilise float et non pas int pour convertir les chiffres car les données peuvent contenir des décimales :
    HPOS_T = float(Alto.xpath("//alto:TextBlock[@ID='{}']/@{}".format(TextBlock_ID, "@HPOS"), namespaces=NS)[0])
    VPOS_T = float(Alto.xpath("//alto:TextBlock[@ID='{}']/@{}".format(TextBlock_ID, "@VPOS"), namespaces=NS)[0])
    WIDTH_T = float(Alto.xpath("//alto:TextBlock[@ID='{}']/@{}".format(TextBlock_ID, "@WIDTH"), namespaces=NS)[0])
    HEIGHT_T = float(Alto.xpath("//alto:TextBlock[@ID='{}']/@{}".format(TextBlock_ID, "@HEIGHT"), namespaces=NS)[0])
```

```

# On calcule quelles TextLignes sont associées à quels TextBlocks vides
# ATTENTION : si les régions n'ont pas été saisies correctement sur escriptorium, cette fonction est susceptible
# de ne pas marcher. Ceci arrive par exemple quand quelqu'un dessine
# une ligne qui dépasse le TextBlock qui est censé le contenir. Cette
# situation s'avérant assez courante, nous avons développé une manière
# de la contourner presque systématiquement en ajoutant une marge
# d'erreur dans le calcul

# on appelle le dictionnaire de zones des lignes et on boucle sur chaque ligne :
for ligne_ID in dic_zones_lignes:
    # on crée des variables signalant les zones de la ligne courante :
    HPOS_L = float(dic_zones_lignes[ligne_ID]["HPOS_L"])
    VPOS_L = float(dic_zones_lignes[ligne_ID]["VPOS_L"])
    WIDTH_L = float(dic_zones_lignes[ligne_ID]["WIDTH_L"])
    HEIGHT_L = float(dic_zones_lignes[ligne_ID]["HEIGHT_L"])
    # On fait les calculs permettant de déterminer si une TextLine est
    # contenue par un TextBlock dans la page :
    # Pour les comprendre, la documentation visuelle des régions iiif peut être utile :
    # https://iiif.io/api/image/2.1/#image-request-parameters
    if HPOS_T <= HPOS_L + 50:
        if VPOS_T <= VPOS_L + 50:
            if HPOS_T + WIDTH_T + 50 >= HPOS_L + WIDTH_L:
                if VPOS_T + HEIGHT_T + 50 >= VPOS_L + HEIGHT_L:
                    # variable signalant la ligne courante :
                    contenu = Alto.xpath("//alto:TextLine[@ID='{}']".format(ligne_ID),
                                         namespaces=NS)[0]
                    # on l'ajoute au TextBlock courant :
                    TextBlock.append(contenu)

```

- 5.1 Extraction des entrées : Maintenant que nous nous sommes assurés que les fichiers ALTO sont correctement structurés, nous pouvons extraire leurs informations et reconstituer les entrées du catalogue. Le script fait appel à une série de fonctions qui traitent chaque ligne, de chaque entrée, de chaque page. Ces fonctions vont utiliser des expressions régulières complexes pour déterminer si la ligne courante est le nom-prénom d'un auteur, une œuvre, ou bien une information complémentaire (adresse, biographie, etc.). Si le travail de correction des transcriptions automatiques et la segmentation manuelles sont correctes, ces fonctions produiront des résultats très satisfaisants.
- 5.2 Après avoir reconstitué chaque entrée de catalogue, ces fonctions les adaptent à l'arbre TEI et l'intègrent dedans. Elles permettent aussi de détecter plusieurs problèmes d'extraction, qui seront indiqués dans le fichier `problemes.txt` afin que l'utilisateur/utilisatrice puisse les corriger facilement (de manière générale, il s'agit de lignes qui n'ont pas été reconnues et qui ont été exclues de l'arbre TEI).
- 6 `output` : TEI et CSV : nous arrivons aux dernières étapes du script. L'arbre TEI est prêt, on va l'écrire dans un fichier déposé dans le chemin `output` indiqué dans la commande. Selon les erreurs de transcription, de segmentation, ou de reconnaissance, ce fichier comportera des erreurs. Il sera utilisé par le script pour produire

deux tableurs CSV, déposés dans un dossier CSV. Selon les erreurs contenues dans le fichier TEI, ces fichiers CSV comporteront des erreurs. Le premier est une version simplifiée adaptée à la lecture humaine et utile pour les corrections manuelles. Le deuxième comporte de nombreuses colonnes et constitue un gabarit pouvant alimenter la base de donnée BasArt du projet ARTl@S.

- 7 Informations à afficher sur le terminal : Finalement, le script construit et affiche des messages d'information ou d'erreur, lesquels ont été produits tout au long du script. Ces messages permettent que l'utilisateur/utilisatrice puisse évaluer simplement le résultat du pipeline. Ils signalent comment corriger les éventuelles erreurs, et affichent des résumés des processus réussis. C'est à travers de cet `output` que l'utilisateur/utilisatrice est susceptible de prendre en main efficacement le programme : il s'agit de produire des explications, même dans les cas où le traitement des fichiers ne soit pas satisfaisant.

6.3 Expressions régulières

L'utilisation d'expressions régulières (regex) pour reconnaître les lignes du catalogue est une étape fondamentale du script. Celles-ci sont implémentées grâce au paquet externe python `re`. Le programme extractionCatalogs est doté de regex adaptées à la très grande majorité des cas possibles, mais il pourrait arriver que l'utilisateur soit mené à devoir les adapter à son catalogue. L'implémentation regex résulte d'un choix pragmatique qui, pour ce cas spécifique, a donné la priorité à l'efficacité face à la pédagogie. Le souhait initial de Béatrice Joyeux-Prunel était que l'interface Jupyter permette que les étudiants indiquent des regex simples adaptées à leurs catalogues. Cependant, ce type de manipulation implique que le script ne soit pas immédiatement fonctionnel, et face à la variété des cas possibles, qui est ample mais semble limitée par la répétition de cas usuels, il a finalement paru préférable de développer des regex longues et solides. Elles s'avèrent difficiles à expliquer à des étudiants, mais pour l'instant très satisfaisantes sur le plan technique. Par ailleurs, les regex étant rassemblées dans un même document `instantiation_regex.py`, il est finalement très simple d'adapter les variables à des cas spécifiques avec des regex courtes personnalisées.

Les catalogues des expositions d'art du XIX^e et XX^e siècle comportent une grande variété de formats de présentation des exposants et de leurs œuvres. Il est possible de dire que chaque catalogue est une configuration unique de conventions qui elles peuvent se répéter souvent. La figure 6.3 présente quelques exemples. Ceux-ci sont inclus dans l'annexe B, qui présente des échantillons de tous les catalogues du corpus Artl@S. Notons qu'on peut ainsi aussi observer des découpes correctes des zones Segmanto `CustomZone:entry`.

Pour l'œil humain, la structure de ces entrées est évidente. Mais la machine est

DUREY (RENÉ), né à Paris. Français. — 4,
Square Desnouettes. S.A.
192. — *Vue sur la Seine*, p.
193. — *Paysage*, p.
194. — *Peinture*, p.

(a) Salon d'automne 1940

DROLLING, né à Paris, ancien pensionnaire de
France à Rome.

24. *Orphée perdant Eurydice*.

Presque aux portes du jour, troublé, hors de lui-même,
Il s'arrête, il se tourne, il revoit ce qu'il aime.
C'en est fait, un coup-d'œil a détruit son bonheur.

(c) Lux 1818

JACOBS, Rodolphe.	HOCHESTTHI, Géralde.
Bruxelles.	Saint-Maur-des-Fossés.
0134. La fille de ferme.	0135. La répétition.
350	350

(e) Strasbourg 1884

DELACROIX (Eugène). — 22, rue de Douai.

69 — *Pieid*.

(g) Rose Croix 1893

DRIVIER (JULIETTE), née à Paris. Française.
— 6, rue Gassendi. S.T.
180 bis. — *Les enfants de Gaston*, p.

(b) Salon d'automne 1940

DAVID (MAXIME), né à Châlons-sur-Marne (Marne), élève
de Mme de Mirbel, chevalier de la Légion-d'Honneur en
1851.
198. Trois portraits d'Abd-el-Kader, représenté sous des
aspects différents (miniatures).
(Salon de 1853.)

(d) Lux 1867

RIMOLI (1941 — S. Paulo) — bp-72

N.º 1, 72, papel.	2.000,00
N.º 2, 72, papel.	2.000,00
N.º 3, 72, papel.	1.000,00

(f) São Paulo 1972

78. — *Paysage des alpes*.

Signé à gauche : G. Courbet.
Sans date.

T. — H. 0.37. L. 0.45.

Appartient à M. de Salverte.

(h) Courbet 1882

FIGURE 6.3 – Exemples d'entrées de catalogue (voir annexe B)

incapable de déchiffrer ces abstractions visuelles. Il serait envisageable d'entrainer des modèles de segmentation pour analyser la disposition spatiale des lignes (différences dans les marges, emphases typographiques), ou même de calculer leurs disposition à partir des informations encodées en ALTO, afin d'intégrer ce facteur à l'extraction. Cependant, ces démarches s'annoncent hasardeuses et ne semblent pour l'instant absolument pas nécessaires, puisque les regex intégrées au script fonctionnent de manière très satisfaisante.

Dès lors que l'espace de l'entrée est découpé dans la page, les informations purement textuelles suffisent pour que le script reconstruire l'entrée de manière structurée. Ces catalogues font des usages très spécifiques et différenciés de signes de ponctuation pour organiser leurs informations : points, tirets, virgules, chiffres, etc. En analysant une entrée ligne par ligne, le script extractionCatalogs détermine à quoi correspond chacune. La première doit normalement présenter un auteur, chose qui est vérifiée avec une regex dédiée. Les lignes suivantes peuvent être des informations complémentaires, donc le script vérifie à partir de quelle ligne les œuvres commencent à être signalées : normalement, ces lignes commencent par des numéros. Mais si le numéro est suivi d'une virgule, il s'agit très probablement d'une adresse, et il existe des cas où les œuvres non sont pas numérotées. Postérieurement, il faut séparer dans une même ligne un exposant ou une œuvre de ses

informations complémentaires. Le script est désormais capable de traiter tous ces facteurs. Le travail de vérification reste important pour corriger des erreurs qui peuvent parfois être récurrentes, voir nombreuses. Très souvent, il s'agit de problèmes dans la transcription manuelle ou automatique (par exemple, une virgule au lieu d'un point, une lettre "O" au lieu du caractère numérique "0" ont un impact décisif sur la performance du programme).

Dans certains cas, il est virtuellement possible qu'un utilisateur ait à intégrer ses propres regex dans le fichier `instantiation_regex.py`, ce qui est relativement facile à faire. Il suffit de changer les variables `exposant_recuperation_regex` et/ou `oeuvre_recuperation_regex`. Au besoin, l'utilisateur peut aussi modifier les autres regex présentes sur le fichier, qui permettent de détecter des adresses, des mesures, des prix, et des informations complémentaires.

Voici le chemin vers ce fichier :

```
extractionCatalogs/extractionCatalogs/variables/instanciation_regex.py
```

Il contient toutes les regex utilisées par le script. Voici les plus importantes :

A [MLE .] * [* [* [a - z à - ü] { 0 , 2 } [.] * [a - z à - ü] { 0 , 2 } [.] * [A - Z À - Ü \ () [,] * [.] * [A - Z C È - Ü a - z o è - ü] ' ' . \ (\) -] + [, .] * [A - Z C È - Ü a - z à - ü] { 0 , 2 } [.] * [V A O N .] * [A - Z C È - Ü \ (\) -] [A - Z C È - Ü a - z o è - ü] ' ' ' \ (\) -] + [.] * [A - Z C È - Ü] * [A - Z C È - Ü a - z o è - ü] ' ' ' . \ (\) -] * [.] * [.] * [^ [A - Z C È - Ü a - z o è - ü] { 0 , 2 } [.] * [A - Z C È - Ü a - z o è - ü] { 0 , 2 } [.] * [A - Z C È - Ü a - z o è - ü] * [. , .] * [.] * [A - Z C È - Ü a - z o è - ü] { 0 , 2 } [.] * [A - Z C È - Ü a - z o è - ü] { 0 , 2 } [.] * [A - Z C È - Ü a - z o è - ü] { 0 , 2 } [.] * [A - Z C È - Ü] + [a - z o è - ü] * [.] * [.] * [, .] * [^ [A - Z C È - Ü] + [a - z o è - ü] * [.] * [, .] *

FIGURE 6.4 – Regex exposant

**^[*]*[.]*\d{1,4}[*].*[.]*([Bis|bis|ter|Ter|BIS|TER])*[.]*[-|^[*]*\d{1,4}]
[-]*([Bis|bis|ter|Ter|BIS|TER])*[.]*[*].[^[*]*\d{1,4}][.]*([Bis|bis|ter|Ter|BIS|TER])*[.]*[-|^[*]*\d{1,4}][.]*[-|^[*]*\d{1,4}][.]*([Bis|bis|ter|Ter|BIS|TER])*[.]*[-]**

FIGURE 6.5 – Regex Oeuvre

Ces regex ont été développées sur Regex101², un site gratuit qui permet de construire, de tester et d'expliquer des expressions régulières. Cet outil permet de sauvegarder des tests, que nous pouvons dès lors présenter en action :

- pour les auteurs : <https://regex101.com/r/FCrEk0/1>
 - Pour les œuvres : <https://regex101.com/r/hZ1iR3/1>

Il faut noter que les exemples d'adresse sont bien écartés au profit des œuvres. Les exemples d'exposants, d'œuvres et d'adresses sont directement issus du corpus de catalogues Artl@s. D'autres regex mineures restent pour autant fondamentales dans le décryptage des lignes et de leurs informations :

2. *Regex101*, URL : <https://regex101.com/>.

- informations complémentaires :
- ```
info_comp_regex = re.compile(r'^(.*$,.*$[^0-9.] -.*$)
```
- adresses : `regex_adresse = re.compile(r'^[*]*\d{1,4}[ bisBIS]*,')`
  - mesures : `regex_mesures = re.compile(r'^[*]*\d{1,4}[ ]*x')`
  - prix : `regex_prix = re.compile(r'^[*]*[0-9]+[\.]*[0-9]* fr[ancs]*[.]*$')`

Si ces expressions régulières permettent de différencier des types d'informations complémentaires, et si elles sont nécessaires pour le fonctionnement du script, celui-ci ne les utilise pas pour encoder ces informations, qui restent sur le niveau de la description. Faute de temps, il n'a pas été possible d'intégrer ces précisions dans la construction de l'arbre TEI, mais le travail est désormais envisageable dans le cadre d'un prochain stage. Il suffira de spécifier des balises où insérer ces informations, et d'adapter les fonctions à cet usage. Par ailleurs, il serait idéal d'intégrer la possibilité d'adapter les regex directement depuis le notebook, en écrivant la variable sur le fichier concerné depuis une cellule. Nous avons fait quelques tests qui, faute de temps, n'ont abouti à rien de concret.

Malgré l'existence de ces horizons suggérés par les directeurs de stage mais restés que superficiellement explorés, et surtout malgré le fait que dans ce cas spécifique l'efficacité technique l'emporte sur la pédagogie, il est important de mentionner que l'un des piliers de l'efficacité du script réside dans les regex actuellement implémentées.

## 6.4 Informations sur le terminal

Lors de l'exécution du script, le terminal (ou la cellule Jupyter Notebook) affiche un certain nombre d'informations. Un exemple a été donné dans le chapitre 6.1 (figure 6.1), et peut être consulté en détail en texte brut dans l'annexe C du mémoire.

pour chaque fichier, le terminal imprimera un récapitulatif avec plusieurs indications :

- Résultats de la vérification de la conformité à l'ontologie Segmonto
- Résultats de restructuration du fichier
- Résultats de la vérification de la segmentation saisie manuellement sur eScriptorium, et résultats de la tentative de correction si des erreurs sont détectées.
- Extraction : affichage des entrées complètes sur le terminal

Suite au traitement de tous les fichiers, et de chaque résumé individuel de page, le terminal indiquera un résumé global :

- Nombre de pages traitées
- Nombre d'entrées extraites
- Nombre d'exposants signalés (normalement, le chiffre est égal à celui des entrées)
- Nombre d'œuvres extraites
- Résultats et problèmes du traitement des fichiers ALTO en input

- Résultats et problèmes du fichier TEI produit
- Réussite ou échec de création des tableurs CSV
- Chemin vers le dossier d'extraction

Pour qu'une extraction fonctionne, il n'est pas nécessaire que le processus soit libre d'erreurs. Le script fait tout ce qui lui est possible, et signale des problème que l'utilisateur pourra corriger selon s'il estime que l'extraction est satisfaisante ou pas.

Une extraction de cinq pages du catalogues de Nancy 1843 est proposée dans le guide Jupyter Notebook comme deuxième exemple avec la commande :

```
!python3 run.py exemples_guide/Cat_Nancy_1843/ exemples_guide/extractions/ Nancy_1843
```

Voici en exemple les messages de terminal relatifs à la première page du catalogue, qui ne contient aucune entrée :

```
1 - Traitement de Cat_Nancy_1843_typo_0000.xml
 1 - Vérification de la conformité à l'ontologie Segmonto
 [!] L'élément TextBlock (ID='eSc_dummyblock_') du fichier 5_560db_default.jpg n'est pas
 conforme à l'ontologie Segmonto.
 [!] Le fichier 5_560db_default.jpg contient un élément TextBlock (ID='eSc_dummyblock_')
 qui n'est pas associé à l'ontologie Segmonto. [voir attribut TAGREFS]
 2 - Restructuration du fichier
 fichier 'Cat_Nancy_1843_typo_0000.xml_restructuration.xml' créé
 3 - Vérification de l'imbrication des zones saisies sur eScriptorium
 les lignes et les zones ont été correctement saisies par l'utilisateur/utilisatrice
 4 - Extraction :
 Ce fichier ne contient pas d'entrées
```

Voici le résumé final de l'extraction :

```
Résumé :
 6 pages de catalogue traitées [fichiers ALTO]
 25 entrées de catalogue extraites [TextBlocks 'CustomZone:entry']
 25 exposants signalés
 75 œuvres extraites

Analyse des fichiers en input [ALTO] :
 [!] 1 fichier non conforme à l'ontologie Segmonto. Consulter le fichier Nancy_1843_problemes.text.

Analyse du fichier en output [TEI] :
 Le document XML produit est conforme au schéma TEI et à l'ODD du projet.

Création d'un tableau csv :
 Le fichier csv a été produit

Chemin du dossier d'extraction : /Users/EstebanSanchez/TNAH_Git/IMAGO-Artl@s/Github_repos/Juliette_Janès/
 IMAGO-Catalogues-Jjanes/extractionCatalogs/exemples_guide/extractions/
 extraction_Nancy_1843
```

Dans ces deux exemples, le terminal signale des erreurs de conformité à l'ontologie Segmonto. Normalement, si l'utilisateur fait une saisie correcte sur l'interface eScriptorium, cela ne devrait pas avoir lieu, mais l'interaction humaine avec la machine doit rester

une source attendue d'erreurs. Dans le catalogue en exemple, ces erreurs n'empêchent pas d'obtenir une extraction très satisfaisante. L'important pour l'utilisateur est d'avoir un retour clair et qu'il soit capable de comprendre ses enjeux. Si l'extraction avait été défaillante, ces informations permettraient de repérer très rapidement les problèmes. Le script permet d'obtenir des informations sur de nombreuses autres types d'erreurs attendues. Une liste exhaustive de ces messages d'erreur peut être consulté en annexe D du mémoire.

## 6.5 Output

La dernière information sur le terminal rappelle le chemin du dossier produit par le script `extractionCatalogs`. Celui-ci contient tous les dossiers et fichiers en `output` :

- `[...].xml` : fichier XML-TEI contenant l'encodage de la totalité des entrées extraites
- `/CSV` : dossier CSV avec deux tableurs :
  - `[...].tableau_simple.csv` : tableau simplifié pour lecture humaine
  - `[...].tableau.csv` : tableau adapté en tant qu'input pour la base de donnée BasArt. Il contient de nombreux champs vides et correspond au gabarit du projet.
- `/restructuration ALTO` : dossier avec des fichiers ALTO "restructurés" "resegmentés". Ils contiennent les corrections des éventuels problèmes de l'`output` de l'interface eScriptorium. Normalement, ces fichiers n'ont aucune utilité directe pour l'étudiant. Ils permettent tout de même d'approfondir l'usage et la compréhension du script : il est possible de faire des vérifications directement sur ces fichiers.
- `[...].problèmes.txt` : fichier de texte brut indiquant des problèmes d'extraction. Ce fichier est produit uniquement dans le cas où le script n'arrive pas à encoder une ou plus de lignes de texte dans l'arbre TEI.

### 6.5.1 Fichier `problèmes.txt`

Le fichier `[...].problèmes.txt` permet de signaler les problèmes d'extraction, et notamment les lignes qui n'ont pas été intégrées à l'`output` TEI. Le script traite chaque ligne, de chaque entrée, de chaque page. Il utilise des regex pour déterminer à quoi correspondent ces lignes, puis les intègre à l'encodage TEI. Il est possible que des lignes ne soient pas reconnues : elles seront ajoutées à ce fichier, avec des indications précises sur leur localisation dans le catalogue (page et ID ou numéro de l'entrée sur le fichier TEI). Le document intégrera aussi les informations générales d'erreur indiquées sur le terminal. Ainsi, l'utilisateur peut localiser très rapidement les erreurs dans l'extraction et les corri-

ger manuellement, que ce soit dans le fichier TEI ou directement dans les tableurs CSV.

Voici un exemple extrait du traitement du catalogue du Salon d'automne 1940<sup>3</sup> :

---

Objets 'entry' avec items non ajoutés au fichier TEI :

- 'LOMBARD (ALFRED) (voir à ALERED LOMBARD).' (entrée 281)
  - 'MALCLES (JEAN-DENIS), né à Paris, Frarçais. - 54, rue Lhomond (Aux Armées). 5.4,D.', "394, bis. - De la, réverie à l'amour", '(composition)' (entrée 297)
- 

Objets 'TextLine' pouvant constituer des lignes de catalogue non ajoutées au fichier TEI

- 54. (page 97)
- 

Fichiers ALTO en input :

- Le fichier 42\_36c57\_default.jpg contient un élément TextBlock (ID='eSc\_textblock\_87e4619f') qui n'est pas associé à l'ontologie Segmonto. [voir attribut TAGREFS]
- 

On peut observer que ce fichier contient plusieurs sections pour organiser les types d'erreurs possibles. La première indique des entrées entières non ajoutées : ce sont donc des listes contenant plusieurs lignes, qui correspondent généralement à des exposants et leurs œuvres associées. La deuxième section indique qu'une balise <TextLine> dans un fichier ALTO n'a pas été traité en tant que ligne du catalogue : l'utilisateur doit déterminer ce qu'il en est. Dans ce cas, après vérification, on observe rapidement sur l'image de la page qu'il s'agit d'une coquille de transcription automatique.

Voici la totalité des sections susceptibles d'apparaître dans le fichier :

- Entrées non intégrées à l'arbre TEI
- Zones Segmonto CustomZone:entryEnd non intégrées à l'arbre TEI
- Lignes non intégrées à l'arbre TEI
- Autres problèmes d'extraction
- Problèmes de conformité Segmonto
- Problèmes liés à l'input ALTO

### 6.5.2 Fichier TEI

L'output XML TEI est un fichier structuré qui contient un encodage de la totalité des entrées du catalogue qui ont été extraites. L'ODD et le schéma ont été développés

---

3. Salon d'automne, Société des artistes décorateurs et Salon des Tuileries, *Catalogue Des Ouvrages de Peinture, Sculpture, Dessin... Exposés Au Palais de Chaillot... Du 3 Avril Au 25 Avril 1940...*

par Caroline Corbières pendant le stage Artl@s 2020, et constituait depuis 2019 l'output final visé par le pipeline.

Le `teiHeader` découle du gabarit nommé `variables/gabarit_teihandler_IMAGO.py` dans le dossier `/variables` et peut-être facilement adapté à tout autre projet de recherche. Il y a un gabarit vide nommé `gabarit_teihandler_vide.py` dans lequel il faut remplir des variables qui correspondent aux balises du `teiHeader`. Postérieurement, il faut indiquer le nom du gabarit produit sur la seule ligne de code du fichier `variables/contenu_teihandler.py`. Ainsi, des étudiants ou des chercheurs travaillant hors du cadre du projet Artl@s peuvent produire très facilement des encodages adaptés à leurs besoins. C'est le cas de Frédérine Corbières, étudiante du certificat en humanités numériques de l'Unige, qui a travaillé sur l'encodage de catalogues pictorialistes de la fin du XIXe siècle, et dont le gabarit `gabarit_teihandler_PictoCatalogs.py` est inclus à titre d'exemple dans le dossier.

Toutes les entrées du catalogue se trouvent dans la balise `<body>` du fichier, à l'intérieur de la balise `<list>`. Une entrée est contenue dans une balise `<entry>`, laquelle est numérotée, possède un attribut ID, et contient un lien vers la découpe de l'image iiif (si celle-ci existe). Une balise `<desc>` (description) contient le nom de l'exposant (balise `<name>`) et ses informations complémentaires (balise `<trait>` et `<p>`). Suivent des balises `<item>`, qui sont numérotées et possèdent un attribut ID. Elles contiennent le numéro de l'œuvre (balise `<num>`), son titre (balise `<title>`) et son éventuelle description (balise `<desc>`).

Voici un exemple d'entrée issue du catalogue Nancy 1843 :

```
<entry n="1" xml:id="Nancy_1843_e1" source="https://gallica.bnf.fr/iiif/ark:/12148/ [...]>
 <desc>
 <name>M. ANDRÉ,</name>
 <trait>
 <p> peintre, rue des Carmes, à Nancy.</p>
 </trait>
 </desc>
 <item n="1" xml:id="Nancy_1843_e1_i1">
 <num>1</num>
 <title>Portrait de M.D.V.....</title>
 <desc>(pastel).</desc>
 </item>
 <item n="2" xml:id="Nancy_1843_e1_i2">
 <num>2</num>
 <title>Idem de M. C. J...., idem.</title>
 <desc>idem.</desc>
 </item>
 <item n="3" xml:id="Nancy_1843_e1_i3">
 <num>3</num>
 <title>L'Aumône, idem.</title>
 </item>
```

```

 <desc>idem.</desc>
</item>
 <item n="4" xml:id="Nancy_1843_e1_i4">
<num>4</num>
<title>Portrait de M. C...., idem.</title>
<desc>idem.</desc>
</item>
 <item n="5" xml:id="Nancy_1843_e1_i5">
<num>5</num>
<title>Idem de M. H....., idem.</title>
<desc>idem.</desc>
</item>
</entry>
```

### 6.5.3 Fichiers CSV

Le stage 2022 a intégrée une dernière étape au pipeline, afin de produire des instruments structurés lisibles par des êtres humains et immédiatement injectables à la base de données BasArt. Il s'agit de produire deux tableurs CSV (Comma Separated Value) contenant la totalité des informations extraites, à partir du fichier TEI. Le script extractionCatalogs utilise pour cela deux feuilles de transformations XSL : `XMLtoCSV.xsl` et `XMLtoCSVsimple.xsl`. La structure de ces feuilles de transformation est issue du travail de Ljudmila Petkovic pour le projet Artl@s en 2019. Il a été question d'adapter la feuille à nos besoins (noms des champs, cellules supplémentaires) et de l'intégrer au script python.

Voici les fichiers produits :

- [...]\_tableau\_simple.csv : Une version simplifiée permet de lire facilement ces informations, afin de vérifier lesquelles doivent être corrigées manuellement.
- [...]\_tableau.csv : Une version complète et adaptée au cahiers des charges de la base de données BasArt du projet Artl@S.

Ce dernier fichier peut alimenter directement la base à travers du site dédié, moyennant une intervention manuelle. L'utilisateur doit l'ouvrir et compléter deux cellules qui se situent en haut à gauche :

<b>Exposition:</b>	COLLER ICI LE NOM DE L'EXPOSITION
<b>Utilisateur:</b>	COLLER ICI LE NOM UTILISATEUR

FIGURE 6.6 – Tableau CSV : cellules à remplir

Les étudiants du certificat en humanités numériques possèdent normalement un compte utilisateur BasArt (<https://artlas.huma-num.fr/fr/bases-en-acces-libre/>) :

- Ils doivent copier/coller ce nom d'utilisateur dans la cellule **COLLER ICI LE NOM UTILISATEUR**

- Puis rechercher le nom de l'exposition/catalogue sur la base, et copier/coller le nom de l'exposition/catalogue dans la cellule **COLLER ICI LE NOM DE L'EXPOSITION**
- Ils peuvent finalement utiliser un formulaire dédié pour envoyer leur tableau CSV

Ces manipulations permettent qu'un travail produit dans un cadre scolaire alimente directement une base de données connue et amplement utilisée comme instrument de recherche dans le milieu de l'histoire de l'art.

## 6.6 Corrections manuelles

Les interventions et les corrections manuelles font partie intégrante du pipeline et peuvent avoir lieu, selon les besoins et les aptitudes de chaque utilisateur, à toutes les étapes du processus. Il est possible d'intervenir sur tous les documents, mais un étudiant débutant pourra se limiter à l'interface eScriptorium et à l'**output CSV**.

- Sur l'interface eScriptorium, l'utilisateur peut produire manuellement ou automatiquement les transcriptions et les segmentations. L'interface permet aussi de faire toutes les corrections nécessaires sur la page d'édition dédiée
- Sur les fichiers XML ALTO produits par eScriptorium, l'utilisateur peut corriger la transcription automatique, le découpage des zones, ainsi que leur conformité à l'ontologie Segmonto. Ces corrections peuvent être faites sur eScriptorium, mais un utilisateur expérimenté gagnera beaucoup de temps en traitant directement les fichiers ALTO
- Le script python extractionCatalogs est librement adaptable aux besoins des utilisateurs. Les manipulations plus simples et attendues sont l'intégration d'expressions régulières au fichier `instanciation_regex.py` et la création de gabarits personnalisés pour le `teiHeader` de projets spécifiques. Un utilisateur avec des connaissances intermédiaires en programmation pourra s'appuyer sur les nombreux commentaires pour modifier le code selon ses besoins
- Sur le fichier XML TEI en **output**, l'utilisateur peut corriger le contenu et la structure des entrées : numéros, texte, attributs, etc. Il pourra ainsi s'assurer d'avoir des encodages bien formés, mais cela n'est pas indispensable pour produire les fichiers CSV
- sur les fichiers CSV en **output**, l'utilisateur peut corriger le contenu et la structure des entrées

Éditer des fichiers XML peut être difficile pour un.e utilisateur/utilisatrice néophyte. Il faut savoir lire un document complexe et veiller à ce que sa conformité soit respectée. Si un logiciel comme Oxygen permet de faire cela de manière efficace, celui-ci est payant et n'est en conséquence pas conforme à un cahier des charges fondé sur les principes de la science ouverte. Notons seulement que les institutions de recherche possèdent souvent

des licences pour utiliser ce logiciel, et que les essais gratuits sont très faciles à obtenir.

L'utilisateur doit utiliser les informations imprimées sur le terminal et sur le fichier [...]\_problèmes.txt pour détecter rapidement les corrections à faire. Il s'agira de corriger des problèmes de transcription, mais aussi d'ajouter des auteurs, des œuvres ou des informations complémentaires exclues pendant l'extraction.

Il est important de noter que les modifications sur les documents XML en `input` (ALTO) et en `output` (TEI) impliqueront que la même commande remplace les résultats antérieurs (si les mêmes chemins et le même titre de catalogue sont indiqués). Quant aux fichiers XML intermédiaires de "restructuration" et de "resegmentation", ils sont produits sur la base des ALTO en `input` et leur modification n'affectera en rien les résultats de l'exécution de la même commande. Cependant, dans ce cas, le terminal affichera un message demandant si l'utilisateur veut remplacer les versions modifiées/anciennes. Sur Jupyter Notebook, toutefois, l'utilisateur ne pourra pas faire ce choix car le message de terminal n'accepte pas d'`input` : il faudra changer systématiquement le nom de catalogue et/ou le chemins d'`output` quand des modifications sont réalisées sur les fichiers ALTO ou TEI. Comme indiqué, cela ne sera pas un problème pour l'usage normal d'un étudiant : seule des interventions plus poussées nécessitent donc de l'utilisation d'un terminal.

Le guide Jupyter Notebook conclut en proposant à l'utilisateur plusieurs exemples de commandes pour transcrire des catalogues. Ils fournissent des cas concrets d'erreurs récurrentes qui ne nuisent pas à l'extraction satisfaisante des entrées. Elles peuvent être lancées directement sur le notebook ou bien sur un terminal `bash`, en se déplaçant sur le dossier `/extractionCatalogs` et en enlevant le signe préalable [!] :

```
!python3 exemples_guide/Cat_Automne_1940/ exemples_guide/extractions/ automne_1940
!python3 run.py exemples_guide/Cat_Refuses_1863/ exemples_guide/extractions/ refuses_1863
!python3 run.py exemples_guide/Cat_Rouen_1853/ exemples_guide/extractions/ rouen_1853
!python3 run.py exemples_guide/Cat_Strasbourg_1884/ exemples_guide/extractions/ Strasbourg_1884
!python3 run.py exemples_guide/PCP_1896/ exemples_guide/extractions/ PCP_1896
!python3 run.py exemples_guide/Cat_SaoPaulo_1972/ exemples_guide/extractions/ saopaulo_1972
```

# Conclusion

Le présent mémoire a proposé un bilan réflexif pour un stage dont les enjeux font écho aux transformations que le numérique opère sur la recherche en sciences humaines. Il a suggéré que ces enjeux, organiquement reliés, étaient à situer sur des plans épistémologiques, institutionnels et pédagogiques. Les questionnements formulés ont permis de comprendre aussi bien que de définir la signification scientifique du travail effectué et sa vocation pédagogique. En tant que synthèse et chronique abordant le processus par ses dimensions intellectuelles, techniques et logistiques, ce mémoire a aussi permis d'élucider le positionnement résolument technicien de la mission, et de la situer dans le paysage académique des humanités numériques.

À partir d'une structure allant du général vers le particulier, nous avons commencé par explorer les enjeux profonds du stage et cheminé vers les aspects purement logistiques et techniques du livrable final. L'introduction présente ses implications scientifiques, techniques et pédagogiques sous un prisme issu de la sociologie et de l'histoire des sciences. Il s'agit premièrement de comprendre les cadres qui donnent sens au stage, et d'exposer les frontières mouvantes d'un travail collaboratif qui implique des chercheurs, des étudiants et des ingénieurs. Puis de réfléchir aux conséquences d'une transposition du catalogue papier vers des supports numériques qui démultiplient les traitements possibles de ses informations et font éclater son unité documentaire. Nous suggérons que ces tensions, à défaut de pouvoir être résolues, permettent de construire des approches rigoureuses.

La première partie décrit les enjeux scientifiques et logistiques du travail en le positionnant dans un historique constitué par quatre stages depuis 2019. Elle commence par une chronique des missions antérieures qui fait état des continuités et des ruptures dans le projet global ; cette synthèse peut servir à des futurs collaborateurs pour définir des directions à prendre à l'avenir. Un premier stage en 2019 a posé les bases avec une réflexion sur l'utilité de l'encodage automatique de catalogues du XIXe siècle. Il a conclu avec une première chaîne de traitement mettant au centre les logiciels Transkribus (reconnaissance de caractères) et Grobid (encodage automatique des entrées). En 2020, le *workflow* a été systématisé pour servir à des corpus plus amples, et une réflexion sur l'encodage des catalogues a abouti à la création d'une ODD solide. En 2021, le pipeline a été transformé dans son ensemble pour transposer les données vers des logiciels conformes à la science ouverte (eScriptorium, kraken, python) : les enjeux étaient la gratuité, la

transparence et la reproductibilité des méthodes. À l'aune du stage 2022, deux défauts étaient manifestes : la complexification de la chaîne au nom de l'efficience la rendait absolument inutilisable par des collaborateurs sans aptitudes techniques préalables ; de plus, ces chaînes impliquaient des interventions conséquentes qui mettaient en cause leur caractère "automatique", voire "semi-automatique". La mission s'est donc tournée vers un objectif éminemment pédagogique, et a consisté à construire un outil simple et solide.

La deuxième partie du mémoire décrit en détail l'utilisation de la chaîne de traitement résultante, et s'attarde dans le fonctionnement du script python qui permet l'encodage des catalogues. Il s'agit d'un pipeline en deux étapes abordables par des néophytes. La première repose dans l'utilisation de l'interface eScriptorium pour produire des transcriptions ALTO à partir des catalogues numérisées. La deuxième se limite à l'exécution du script extractionCatalogs à travers d'un notebook Jupyter, qui sert d'interface d'utilisation en même temps que de guide. Les interventions intermédiaires se réduisent au téléchargement de fichiers et à leur positionnement dans des dossiers souhaités. Nous considérons que cela constitue une amélioration conséquente face aux conditions antérieures, qui impliquaient de nombreuses étapes médiées par des interventions hasardeuses (utilisation du terminal, commandes confuses, transformations XSL des fichiers, prise en main de logiciels complexes, instructions difficiles). Un autre apport réside dans l'éloquence du script : outre le fait qu'il soit amplement documenté (dans le mémoire, dans le notebook et dans les fichiers .py), il produit des message de réussite et d'erreur conçus pour que l'utilisateur comprenne les résultats, même quand ceux-ci ne sont pas satisfaisants.

Le pipeline semi-automatique extractionCatalogs (océrisation avec eScriptorium + extraction avec le script extractionCatalogs), accompagné par une interface et guide d'utilisation (Jupyter notebook), a donc un intérêt scientifique (produire des données pour la recherche), technique (produire efficacement ces données) et pédagogique (présenter des technologies numériques fondamentales dans le domaine des humanités numériques). Nous estimons que la priorité donnée à l'enseignement face à l'efficacité ne remet pas en question l'utilité pragmatique du pipeline. Pour la première fois, Artl@s compte avec un outil qui peut être véritablement déployé par ses collaborateurs, et il est virtuellement possible que cela ait un impact sur l'enrichissement de la base de données BasArt.

Au delà, il reste à faire un bilan concret sur l'instrument développé, avec son utilisation dans le cadre des cours d'initiation aux humanités numériques de Béatrice Joyeux-Prunel. Nous sommes invités à intervenir à l'université de Genève pendant l'année scolaire 2022-2023 pour accompagner les étudiants dans l'utilisation du pipeline. Cela nous permettra de contribuer à l'évaluation de l'outil, voire à une définition des problématiques possibles pour la continuation du projet avec un stage en 2023. Nous pouvons rappeler pour l'instant, à travers de l'image de "l'ingénieur lettré", que notre rôle a impliqué de doubler au développement de nos aptitudes numériques une compréhension profonde des

enjeux scientifiques et pédagogiques du projet. Nous pouvons dès lors chercher les angles morts du stage 2022, et nous demander s'il existe des solutions plus efficaces pour équilibrer les pôles différenciés que constituent ses objectifs. Nous pourrions commencer par interroger ces instruments nouveaux pour savoir s'ils imposent des pertes, des lacunes, et ne pas oublier que les catalogues traités, en tant qu'unités intellectuelles et matérielles, constituent des traces historiques lourdes en significations face auxquelles le numérique ne saurait se substituer.



## Annexe A

### liste des améliorations dans le script python

Cette liste concerne des améliorations ou des corrections effectuées sur le prototype de script python de 2021.

- Exclusion des fichiers autres que ALTO et des fichiers/dossiers cachés dans le dossier en `input` (permet entre autres l'utilisation de MacOS avec l'omission du dossier `.DS_Store`). Cela permet d'avoir des dossiers contenant des images ou d'autres types de fichiers
- Exclusion des fichiers ALTO restructurés : la commande peut être exécutée plusieurs fois en évitant la multiplication en boucle des restructurations
- Tabulations et précisions sur le terminal pour rendre les messages plus lisibles avec une visualisation structurée des entrées
- Le fichier TEI en `output` est correctement indenté. Il contient des commentaires pour guider la saisie du `teiHeader`
- `teiHeader` produit en avec la librairie `lxml` est désormais conforme (l'arbre produit contenait plusieurs erreurs d'imbrication)
- simplification du fichier de création `teiHeader` : variables renommées, refactorisées, réordonnées
- restructuration et refactorisation des dossiers du script dans le dossier `/extractionCatalogs` pour plus de clarté
- Ajout de commentaires nombreux et détaillés sur le script pour guider l'utilisateur dans un but pédagogique
- Réécriture du README et des commentaires déjà existants pour plus de clarté
- Création de gabarits interchangeables pour les information fondamentales du `teiHeader` (dossier `/variables`)
- les fichier restructurés sont désormais placés dans un dossier dédié `restructuration`, et non plus dans le dossier des fichiers en `input`

- Lorsqu'un des ALTO en `input` a déjà subi une restructuration, le terminal averti que le fichier produit sera pareil et demande si l'on veut continuer. Si ce n'est pas le cas, le fichier n'est pas restructuré une deuxième fois, et l'ALTO en `input` déjà structuré est utilisé à sa place
- si le catalogue est issu d'un manifeste iiif : le terminal affiche un lien vers une découpe des zones de chaque entrée du catalogue
- Il n'est plus nécessaire d'intervenir sur le code pour désactiver les fonctions relatives aux zones Segmonto `CustomZone:entryEnd` si le catalogue n'en a pas
- le pipeline ignore les régions vides dans la transcription ALTO, qui sont souvent des erreurs de segmentation sur eScriptorium. Cela permet de ne pas générer des entrées vides qui nuisent à la conformité et à la structure de l'`output` TEI (la numérotation résultait inexacte).
- Lorsque la segmentation d'un fichier n'est pas conforme à l'ontologie Segmonto, un message sur le terminal l'indique et renvoie vers le répertoire Segmonto
- Le terminal indique si un fichier traité ne contient pas d'entrées (c'est par exemple le cas pour une page de titre)
- Le terminal affiche le nombre total d'entrées extraites. Si aucune entrée n'est extraite des fichiers restructurés, le terminal affiche qu'il s'agit peut être d'une erreur de construction des expressions régulières ou de conformité à l'ontologie Segmonto
- les noms des fichiers ALTO peuvent avoir ou ne pas avoir des "zéros non significatifs" (ex : "9" ou "09"), ils seront traités dans tous les cas dans l'ordre correct
- le fichier de problèmes `[...].problèmes.txt` indique maintenant si les items sont des `CustomZone:entryEnd` ou des `CustomZone:entry`
- le terminal comporte un résumé qui indique :
  - nombre d'entrées extraites
  - nombre d'auteurs signalés
  - nombre d'auteurs non signalés (et ID des entrées concernées)
  - nombre d'œuvres extraites
- la commande "-v" pour vérifier la conformité des fichiers ALTO est maintenant fonctionnelle. Elle vérifie si une une `TextLine` se trouve directement dans un `TextBLock` associé à une zone Segmonto `MainZone` et non pas à une `CustomZone:entry` (ce problème est inhérent à eScriptorium et rend l'extraction impossible). La commande affiche aussi s'il y a des erreurs de conformité ALTO ou à l'ontologie Segmonto
- Les items avec plusieurs lignes sont traités correctement
- L'utilisateur n'a plus besoin d'intervenir sur le script pour adapter les expressions régulières à son catalogue. Les regex implémentées sont normalement capables de gérer la très grande majorité des cas possibles
- Les adresses sont correctement différencierées des items
- Items "bis" pris en compte pour ne pas nuire à la conformité TEI

- le document [...]\_problèmes.txt est désormais divisé en parties : auteurs non signalés, `CustomZone:entry` non ajoutées, `CustomZone:entryEnd` non ajoutées, récapitulatif général des erreurs
- Dans l'output TEI, les balises de description `<trait>` et `<p>` ne sont créés que si l'entrée contient une description
- Commandes obligatoires de définition du type du catalogue éliminées : "Nulle", "Triple", "Double", "Simple" : le script détecte automatiquement le type de catalogue
- Le terminal affiche maintenant les descriptions des œuvres
- Le script capable de reconnaître des œuvres dans des catalogues où ceux-ci ne sont pas numérotés
- Étape supplémentaire : le script opère deux transformations XSL et produit deux tableurs CSV à partir du fichier TEI



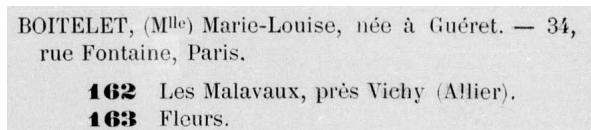
## Annexe B

### Échantillons d'entrées de catalogue

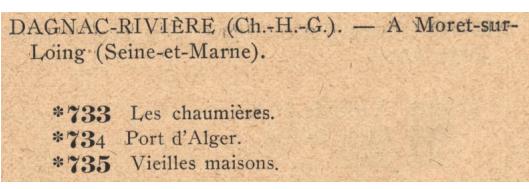
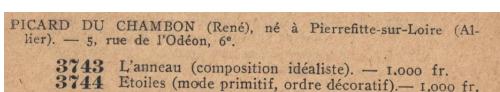
Chaque échantillon a été sélectionné au hasard dans chacun des catalogues du corpus traité. En résultent des exemples très représentatifs, avec quelques exceptions que le script extractionCatalogs est capable de traiter correctement.

La découpe de l'image correspond à la région `CustomZone:entry` de l'ontologie Segmonto.

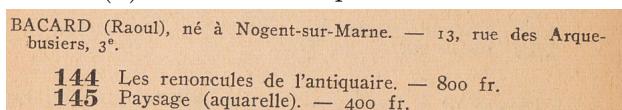
Les références complètes de chaque catalogue peuvent être trouvées dans la Bibliographie, dans la section "Catalogues traités (corpus Artl@s)". L'objectif de cet annexe est d'offrir des repères visuels généraux au lecteur, et la sélection n'a pas suivi aucun protocole particulier d'échantillonnage.



(a) Salon des Indépendants 1892



(b) Salon des Indépendants 1913



(c) Salon des Indépendants 1923

(d) Salon des Indépendants 1935

FIGURE B.1 – Salon des Indépendants

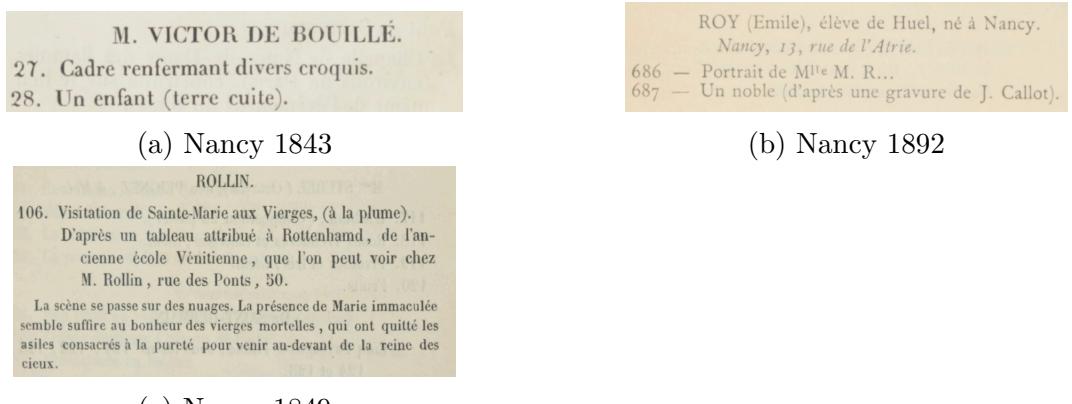


FIGURE B.2 – Société lorraine des amis des arts

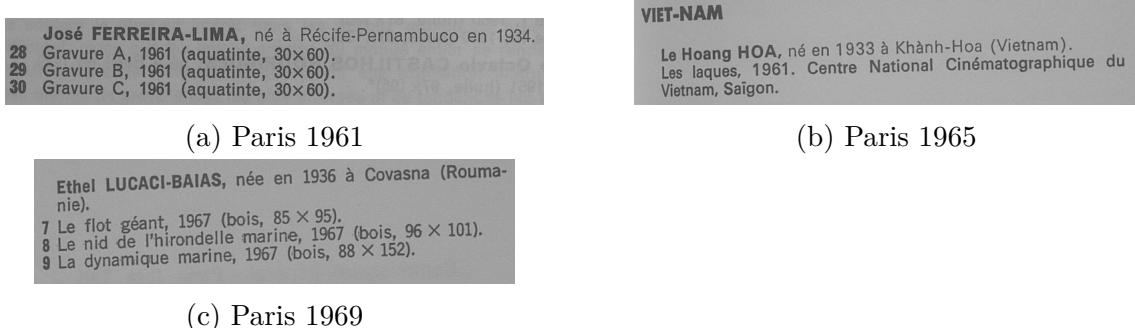
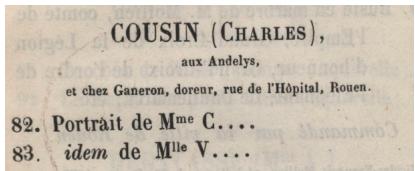
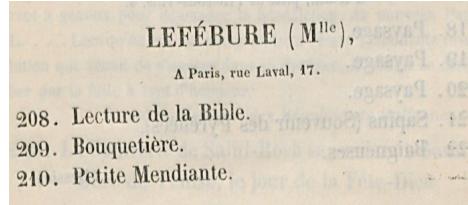


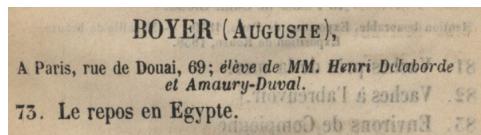
FIGURE B.3 – Paris



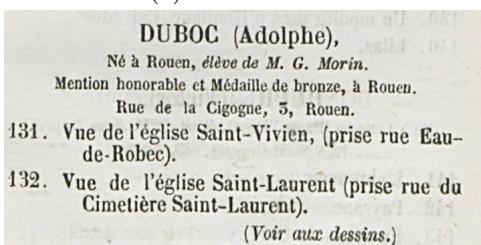
(a) Rouen 1853



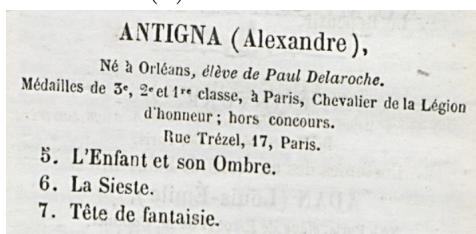
(b) Rouen 1856



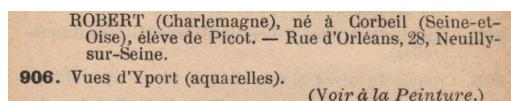
(c) Rouen 1860



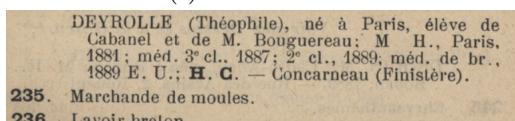
(d) Rouen 1869



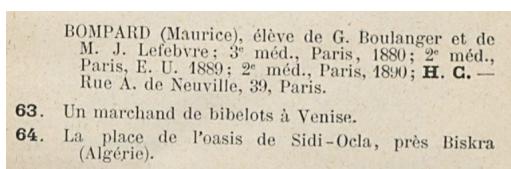
(e) Rouen 1872



(f) Rouen 1878



(g) Rouen 1888



(h) Rouen 1891

**Edward HOPPER (EE. UU. 1882 —)**

- 29. Posto de gasolina — 1940. 67x102. Fundo Sra. Simon Guggenheim.
- 30. Madrugada em Pennsylvania — 1942. 62x112. Sr. e Sra. Otto L. Spaeth, New York
- 31. Manhã no Cabo Cod — 1950. 86,3x102. Frank K.M. Hehn Galleries, New York

(i) Rouen 1895

(j) Rouen 1851

FIGURE B.4 – Exposition annuelle du musée de Rouen

**Dalbono** Eduardo. — N. a Napoli nel 1843; cominciò i suoi studi a Roma, indi tornato nella sua città, ebbe a maestri Domenico Morelli e il Mancinelli. Dimorò poi otto anni a Parigi. Fra le numerose opere uscite dal suo pennello, una delle più felici è « *La leggenda delle Sirene* ». Egli rende con foga di colore e d'immaginazione l'intensa azzurrità del suo mare.

**74 Il mare a Torre Annunziata.**

(a) Venezia 1895

**Soudbinine Seraphin.**

65 *La danza* (legno).

66 *La Vergine* (ebano).

67 *Testa decorativa* (cemento).

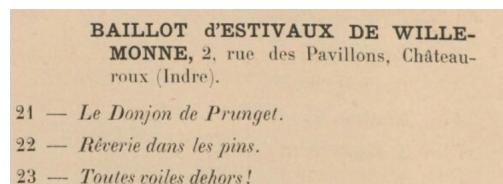
(c) Venezia 1920

**Magyar-Mannheimer** Gusztáv

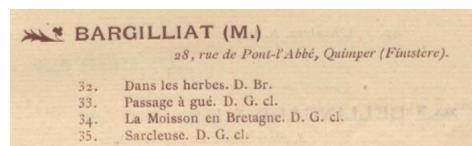
**13 Mattino sul lago di Balaton** (appartiene allo Stato ungherese).

(b) Venezia 1905

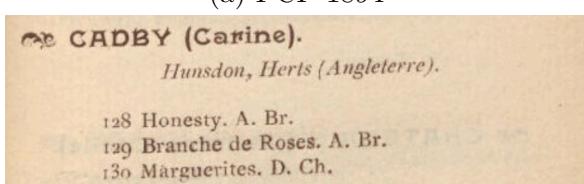
FIGURE B.5 – Bienale de Venise



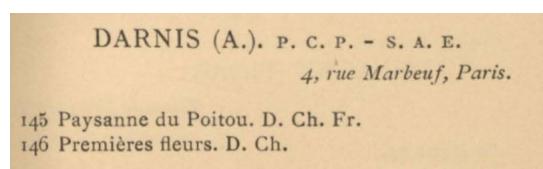
(a) PCP 1894



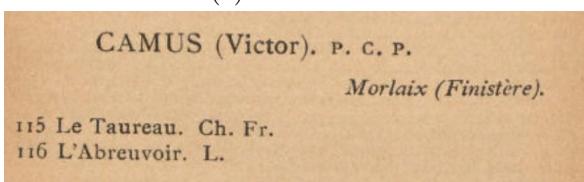
(b) PCP 1896



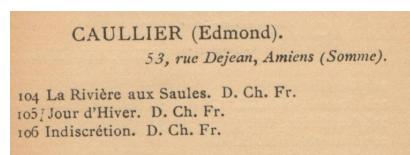
(c) PCP 1898



(d) PCP 1902



(e) PCP 1904



(f) PCP 1906

FIGURE B.6 – Photo Club Paris (PCP)

DUREY (RENÉ), né à Paris. Français. — 4,  
Square Desnouettes. S.A.

**192.** — *Vue sur la Seine*, p.

**193.** — *Paysage*, p.

**194.** — *Peinture*, p.

(a) Salon d'automne 1940

DROLLING, né à Paris, ancien pensionnaire de France à Rome.

24. Orphée perdant Eurydice.

Presque aux portes du jour, troublé, hors de lui-même, Il s'arrête, il se tourne, il revoit ce qu'il aime. C'en est fait, un coup-d'œil a détruit son bonheur.

(c) Lux 1818

JACOBS, Rodolphe. Bruxelles. 0134. La fille de ferme. 0135. La répétition. 0350

(e) Strasbourg 1884

DELACROIX (Eugène). — 22, rue de Douai.

69 — *Picidé*.

(g) Rose Croix 1893

DRIVIER (JULIETTE), née à Paris. Française. — 6, rue Gassendi. S.T.

**180 bis.** — *Les enfants de Gaston*, p.

(b) Salon d'automne 1940

DAVID (MAXIME), né à Châlons-sur-Marne (Marne), élève de M<sup>e</sup> de Mirbel, chevalier de la Légion-d'Honneur en 1851.

**198.** Trois portraits d'Abd-el-Kader, représenté sous des aspects différents (miniatures). (Salon de 1853.)

(d) Lux 1867

RIMOLI (1941 — S. Paulo) — bp-72

N. <sup>o</sup> 1, 72, papel.	2.000,00
N. <sup>o</sup> 2, 72, papel.	2.000,00
N. <sup>o</sup> 3, 72, papel.	1.000,00

(f) São Paulo 1972

78. — *Paysage des alpes*.

Signé à gauche : G. Courbet.  
Sans date.

T. — H. 0.37, L. 0.45.  
Appartient à M. de Salverte.

(h) Courbet 1882

FIGURE B.7 – Catalogues non groupés



## Annexe C

# Informations de terminal pour une page de catalogue extraite

L'exemple ci dessus est issu de la première page contenant des entrées du catalogue du Salon de Nancy 1843.<sup>1</sup>

```
=====
1 - Traitement de Cat_Nancy_1843_typo_0001.xml
 1 - Vérification de la conformité à l'ontologie Segmonto
 le fichier issu d'eScriptorium est conforme à l'ontologie Segmonto
 2 - Restructuration du fichier
 fichier 'Cat_Nancy_1843_typo_0001.xml_restructuration.xml' créé
 3 - Vérification de l'imbrication des zones saisies sur eScriptorium
 Le Ficher Alto produit par eScriptorium contient des zones ['TextBlock'] vides
 - Parmi ces zones ['TextBlock'] vides, il y a des 'CustomZone:entry'
 Correction automatique : 18 lignes correspondant à 2 zones vides ont été déplacées vers ces zones
 4 - Extraction :
```

M. ANDRÉ,  
peintre, rue des Carmes, à Nancy.  
1. Portrait de M.D.V.....  
 (pastel).  
2. Idem de M. C. J...., idem.  
 idem.  
3. L'Aumône, idem.  
 idem.  
4. Portrait de M. C...., idem.  
 idem.  
5. Idem de M. H....., idem.  
 idem.  
6. La Sorcière. idem.  
7. Portrait de M.....  
 (pastel.)  
Image : https://gallica.bnf.fr/iiif/ark:/12148/bpt6k62251805/f7/  
217.0,1320.0,1333.0,735.0/full/0/default.jpg  
M. ANDRÉ JEUNE,  
professeur de dessin de machines, 11, rue de la Pépinière.

---

1. Société lorraine des amis des arts, *Catalogue Des Peintures, Miniatures, Aquarelles, Dessins, Sculptures et Lithographies Exposés à Nancy... Par Les Artistes Lorrains...*

8. Locomotive  
(dessin au lavis).  
9. Machine, idem.  
idem.  
10. Pont de Suresnes.  
11. La Chapelle de Notre-de-Dame des flammes même de  
l'évènement du 8 mai.  
Image : <https://gallica.bnf.fr/iiif/ark:/12148/bpt6k62251805/f7/203.0,2079.0,1354.0,714.0/full/0/default.jpg>

=====

Résumé :

1 page de catalogue traitée [fichier ALTO]  
2 entrées de catalogue extraites [TextBlocks 'CustomZone:entry']  
2 exposants signalés  
11 œuvres extraites

Analyse des fichiers en input [ALTO] :

La segmentation des pages est conforme à l'ontologie Segmonto

Analyse du fichier en output [TEI] :

Le document XML produit est conforme au schéma TEI et à l'ODD du projet.

Création d'un tableau csv :

Le fichier csv a été produit

Chemin du dossier d'extraction : /Users/EstebanSanchez/TNAH\_Git/IMAGO-Art1@s/Github\_repos/Juliette\_Janès/IMAGO-Catalogues-Jjanes/extractionCatalogs/exemples\_guide/extractions/extraction\_Nancy\_1843\_page

## Annexe D

### Liste des erreurs affichables par le terminal

- [!] ATTENTION : ce fichier avait déjà été restructuré ; le nouveau fichier produit est identique."
- [!] ATTENTION : un fichier TEI avec le même nom de catalogue a déjà été produit et diffère dans son contenu; Soit ce fichier a été MODIFIÉ/CORRIGÉ MANUELLEMENT, soit les fichiers ALTO en input ont été MODIFIÉS/CORRIGÉS MANUELLEMENT."
- [!] ATTENTION : Aucune entrée n'a été extraite des fichiers restructurés ; veuillez vérifier les regex utilisées pour l'extraction ou la structure des fichiers ALTO en input"
- [!] [...] lignes non traitées font potentiellement partie des entrées du catalogue. Elles ont été ajoutées au fichier [...]\_problems.txt
- [!] [...] fichiers non conformes à l'ontologie Segmonto. Consulter le fichier [...]\_problemes.text.
- [!] [...] entrées de catalogue [TextBlock 'CustomZone:entry'] n'ont pas été intégrées au fichier TEI. Elles ont été ajoutées au fichier [...]\_problems.txt
- [!] [...] fins d'entrée de catalogue [TextBlock 'CustomZone:entryEnd'] n'ont pas été intégrées au fichier TEI. Elles ont été ajoutées au fichier [...]\_problems.txt
- [!] Le fichier csv n'a pas été produit. Vérifiez la conformité du fichier TEI.
- exposants non signalés : [...]
- [!] L'entryEnd suivante n'a pas été extraite : [...]
- ATTENTION : Ce fichier ALTO a été restructuré, mais il n'est pas conforme à l'ontologie Segmonto. Les entrées doivent être indiquées comme des 'CustomZone:entry' lors de la segmentation. Vous pouvez changer manuellement

- dans les fichier ALTO le nom du type de zone correspondant (balise <OtherTag LABEL='''>). Pour plus d'informations : <https://github.com/SegmOnto>
- [!] ATTENTION : Les lignes suivantes ne sont associées à aucune entry/entryEnd et n'ont pas été traitées [Les TextLine se situent directement dans des TextBlock 'Mainzone']
  - Sections dans le fichier [...]problèmes.txt :
    - Objets 'entry' avec items non ajoutés au fichier TEI : [...]
    - Objets 'entryEnd' non ajoutés au fichier TEI : [...]
    - Objets 'TextLine' pouvant constituer des lignes de catalogue non ajoutées au fichier TEI : [...]

# Table des figures

1.1	Chaîne de traitement en 2019 . . . . .	14
1.2	Chaîne de traitement en 2020 . . . . .	18
1.3	Typologie des catalogues pour le programme python en 2021 . . . . .	22
1.4	Chaîne de traitement en 2021 . . . . .	22
3.1	<code>input</code> : exemple de page de catalogue (Salon d'automne 1940) . . . . .	41
3.2	<code>output</code> : tableau contenant les informations de la page de catalogue . . . . .	42
4.1	Activation de la table des matières et des sections collapsables . . . . .	48
4.2	Table des matières du notebook . . . . .	49
4.3	Diminuer une cellule sur le notebook . . . . .	49
5.1	eScriptorium : créer un projet . . . . .	53
5.2	eScriptorium : créer un projet (2) . . . . .	54
5.3	eScriptorium : créer un projet (3) . . . . .	54
5.4	eScriptorium : créer un projet (4) . . . . .	54
5.5	eScriptorium : créer un projet (5) . . . . .	55
5.6	Salon de Nancy 1843, p. 6. . . . .	56
5.7	eScriptorium : Saisie des zones Segmonto . . . . .	57
5.8	Icône iiif . . . . .	58
5.9	iiif sur Gallica . . . . .	59
5.10	Manifeste iiif sur Gallica . . . . .	59
5.11	eScriptorium : choisir des images . . . . .	60
5.12	eScriptorium : frise d'images . . . . .	60
5.13	eScriptorium : icône image . . . . .	61
5.14	eScriptorium : sélection d'images dans la frise . . . . .	62
5.15	eScriptorium : reconnaissance des lignes . . . . .	62
5.16	eScriptorium : lancer la transcription . . . . .	63
5.17	eScriptorium : page d'édition . . . . .	63
5.18	eScriptorium : corrections manuelles . . . . .	64
5.19	eScriptorium : lignes avec erreurs . . . . .	65
5.20	eScriptorium : saisir des zones . . . . .	65

5.21	eScriptorium : MainZone . . . . .	66
5.22	eScriptorium : télécharger les fichiers ALTO . . . . .	67
5.23	eScriptorium : modalités d'export . . . . .	67
5.24	eScriptorium : Download . . . . .	68
5.25	ALTO : image liée . . . . .	68
5.26	ALTO : Description . . . . .	69
5.27	ALTO : Tags . . . . .	69
5.28	ALTO : Layout . . . . .	69
6.1	Script : premier exemple d'extraction . . . . .	73
6.2	Script : gabarit personnalisé pour le teiHeader . . . . .	77
6.3	Exemples d'entrées de catalogue (voir annexe B) . . . . .	81
6.4	Regex exposant . . . . .	82
6.5	Regex Oeuvre . . . . .	82
6.6	Tableau CSV : cellules à remplir . . . . .	88
B.1	Salon des Indépendants . . . . .	99
B.2	Société lorraine des amis des arts . . . . .	100
B.3	Paris . . . . .	100
B.4	Exposition annuelle du musée de Rouen . . . . .	101
B.5	Bienale de Venise . . . . .	102
B.6	Photo Club Paris (PCP) . . . . .	102
B.7	Catalogues non groupés . . . . .	103

# Table des matières

<b>Résumé</b>	<b>i</b>
<b>Remerciements</b>	<b>iii</b>
<b>Bibliographie</b>	<b>v</b>
<b>Introduction : entre enjeux techniques, scientifiques et pédagogiques</b>	<b>1</b>
l'ingénieur, le chercheur et l'étudiant face aux humanités numériques . . . . .	3
Vers une histoire globale des circulations artistiques à travers des humanités numériques . . . . .	5
<b>I Objectifs scientifiques et défis techniques de la mission</b>	<b>9</b>
<b>1 Poursuivre un projet</b>	<b>11</b>
1.1 Stage 2019 : premières réflexions et chaîne de traitement (Transkribus + Grobid) . . . . .	12
1.2 Stage 2020 : systématiser un workflow (Transkribus + Grobid + ALTO) . .	15
1.3 Stage 2021 : migration vers la science ouverte (kraken + python) . . . . .	18
<b>2 Stage 2022 : produire efficacement des données ou enseigner correctement ?</b>	<b>23</b>
2.1 Positionnement de la mission . . . . .	24
2.2 Projets exclus . . . . .	25
2.3 Le choix d'un pipeline échelonné et pédagogique . . . . .	26
<b>3 Résultats du travail</b>	<b>29</b>
3.1 Problèmes techniques affrontés pendant la mission . . . . .	30
3.2 Un instrument efficace et éloquent . . . . .	32
3.3 Améliorations possibles . . . . .	34

<b>II Fonctionnement du pipeline "extractionCatalogs"</b>	<b>37</b>
<b>4 Installation : dépôt et Jupyter notebook</b>	<b>45</b>
4.1 Installer le dépôt : alternatives . . . . .	45
4.2 Installer et configurer le guide : Jupyter notebook et nbextensions . . . . .	46
4.2.1 Jupyter Notebook . . . . .	46
4.2.2 nbextension : extensions pour Jupyter Notebook . . . . .	47
<b>5 Reconnaissance de caractères : interface eScriptorium</b>	<b>51</b>
5.1 OCR avec eScriptorium et Kraken . . . . .	51
5.2 Mettre en place un projet sur eScriptorium . . . . .	53
5.2.1 Créer un projet . . . . .	53
5.2.2 L'Ontologie Segmonto . . . . .	54
5.2.3 Quelles images utiliser ? Le standard iiif et les fichiers images classiques (jpeg, png, pdf) . . . . .	58
5.2.4 Ajouter des images (iiif, jpeg, png, pdf, etc.) . . . . .	60
5.3 Transcription automatique des images . . . . .	61
5.4 Segmentation manuelle des images . . . . .	64
5.5 Output ALTO . . . . .	67
<b>6 Utiliser le script python "extractionCatalogs"</b>	<b>71</b>
6.1 Commandes du script . . . . .	72
6.2 Fonctionnement du script . . . . .	74
6.3 Expressions régulières . . . . .	80
6.4 Informations sur le terminal . . . . .	83
6.5 Output . . . . .	85
6.5.1 Fichier problèmes.txt . . . . .	85
6.5.2 Fichier TEI . . . . .	86
6.5.3 Fichiers CSV . . . . .	88
6.6 Corrections manuelles . . . . .	89
<b>Conclusion</b>	<b>91</b>
<b>A liste des améliorations dans le script python</b>	<b>95</b>
<b>B Échantillons d'entrées de catalogue</b>	<b>99</b>
<b>C Informations de terminal pour une page de catalogue extraite</b>	<b>105</b>
<b>D Liste des erreurs affichables par le terminal</b>	<b>107</b>