



Ecole Polytechnique de l'Université de Tours
Département Informatique
64 avenue Jean Portalis
37200 Tours, France
Tél. +33 (0)2 47 36 14 14
polytech.univ-tours.fr

**Projet Recherche & Développement
2021-2022**

Outils pour l'exploitation de la prosopographie dynamique: ontologie et visualisations



**POLYTECH[®]
TOURS**

Entreprise

Polytech



Tuteur entreprise

Elena PIERAZZO

Étudiant

Olivier MILLOCHAU (DI5)

Tuteur académique

Mohamed SLIMANE

27 mars 2022

Liste des intervenants

Entreprise

Polytech
64 avenue Jean Portalis
37200 Tours, France
polytech.univ-tours.fr



Nom	Email	Qualité
Olivier MILLOCHAU	olivier.millochau@etu.univ-tours.fr	Étudiant DI5
Mohamed SLIMANE	emmohamed.slimane@univ-tours.fr	Tuteur académique, Département Informatique
Elena PIERAZZO	email@univ-tours.fr	Tuteur entreprise



Avertissement

Ce document a été rédigé par Olivier MILLOCHAU susnommé l'auteur.

L'entreprise Polytech est représentée par Elena PIERAZZO susnommé le tuteur entreprise.

L'Ecole Polytechnique de l'Université de Tours est représentée par Mohamed SLIMANE susnommé le tuteur académique.

Par l'utilisation de ce modèle de document, l'ensemble des intervenants du projet acceptent les conditions définies ci-après.

L'auteur reconnaît assumer l'entière responsabilité du contenu du document ainsi que toutes suites judiciaires qui pourraient en découler du fait du non respect des lois ou des droits d'auteur.

L'auteur atteste que les propos du document sont sincères et assume l'entière responsabilité de la véracité des propos.

L'auteur atteste ne pas s'approprier le travail d'autrui et que le document ne contient aucun plagiat.

L'auteur atteste que le document ne contient aucun propos diffamatoire ou condamnable devant la loi.

L'auteur reconnaît qu'il ne peut diffuser ce document en partie ou en intégralité sous quelque forme que ce soit sans l'accord préalable du tuteur académique et de l'entreprise.

L'auteur autorise l'école polytechnique de l'université François Rabelais de Tours à diffuser tout ou partie de ce document, sous quelque forme que ce soit, y compris après transformation en citant la source. Cette diffusion devra se faire gracieusement et être accompagnée du présent avertissement.



Pour citer ce document

Olivier MILLOCHAU, *Outils pour l'exploitation de la prosopographie dynamique: ontologie et visualisations*, Projet Recherche & Développement, Ecole Polytechnique de l'Université de Tours, Tours, France, 2021-2022.

```
@mastersthesis{
  author={MILLOCHAU, Olivier},
  title={Outils pour l'exploitation de la prosopographie dynamique: ontologie et visualisations},
  type={Projet Recherche \& Développement},
  school={Ecole Polytechnique de l'Université de Tours},
  address={Tours, France},
  year={2021-2022}
}
```

Table des matières

Liste des intervenants	a
Avertissement	b
Pour citer ce document	c
Table des matières	i
Table des figures	iv
1 Introduction	1
1 Enjeux et Contexte.....	1
2 Objectifs.....	2
3 Organisation.....	2
2 Description générale	4
1 API.....	4
1.1 Environnement du projet.....	4
1.2 Caractéristiques des utilisateurs.....	4
1.3 Fonctionnalités du système.....	4
1.4 Structure générale du système.....	5
2 Outils de visualisation.....	5
2.1 Environnement du projet.....	5
2.2 Caractéristiques des utilisateurs.....	6
2.3 Fonctionnalités du système.....	6
2.4 Structure générale du système.....	6

3	État de l'art / Veille technologique	7
1	Write.control	7
	État civil.....	8
	Physique	8
	Caractère	9
	Profil.....	10
	Evolution	11
1.1	Analyse critique.....	12
2	Voyant.....	12
	Cirrus : un nuage de mot.....	13
	Lecteur : Le corpus de texte	14
	Tendances	15
	Résumés.....	17
	Contextes.....	18
2.1	Analyse critique.....	18
3	LinkedIn Maps.....	19
3.1	Analyse critique.....	20
4	Analyse et conception	21
1	API	21
1.1	Analyse	21
1.2	Spécifications.....	21
1.2.1	Fonction 1 : getOneCharacter	22
1.2.2	Fonction 2	22
2	Outils de visualisations	22
2.1	Analyse	22
2.1.1	Spécifications.....	23
2.1.2	Fonction 1 : drawRelationshipsNetwork	23
2.1.3	Fonction 2 : drawPsychologicalStatus	23
5	Mise en œuvre	24
1	Déviation par rapport aux spécifications	24
2	1er développements.....	24
2.1	Analyse syntaxique	24
3	2nd développements.....	25
3.1	Communications des données.....	25
3.2	Ecran de sélection.....	26
4	3eme développements.....	26
4.1	Graphe de l'évolution psychologique des personnages	27

6	Bilan et conclusion	34
1	Bilan du semestre 9	34
2	Bilan sur la qualité	34
3	Bilan autocritique sur la gestion	35
Annexes		36
A	Gestion de projet	37
1	Planification	37
2	Description des tâches.....	37
	Tâche 1 : Mise au point ontologie	37
	Tâche 2 : Prise en main outils API	38
	Tâche 3 : Développement de l'API.....	38
	Tâche 4 : Prise en main des outils de visualisations	38
	Tâche 5 : Développement des interfaces de sélection des graphes.....	38
	Tâche 6 : Développement de l'affichage du graphe du réseau social ..	38
	Tâche 7 : Développement de l'affichage du graphe de l'état psycho- logique	39
	Tâche 8 : Rédaction des documents projet	39
B	Cahier de Spécifications	40
C	Bibliographie	59



Table des figures

2 Description générale

2.1 Schéma de l'architecture et du fonctionnement général du système.....	5
---	---

5 Mise en œuvre

5.1 L'écran de sélection de textes et d'un graphe	26
5.2 Première version du graphe de l'état psychologique	27
5.3 Seconde version du graphe de l'état psychologique	28
5.4 Troisième version du graphe de l'état psychologique	29
5.5 Quatrième version du graphe de l'état psychologique.....	30

A Gestion de projet

A.1 Le diagramme de Gantt initial prévu pour le semestre 10.....	37
A.2 Le diagramme de Gantt final à la fin du semestre 10.....	37

1

Introduction

Ce document présente la réalisation du projet recherche et développement « Outils pour l'exploitation de la prosopographie dynamique : ontologie et visualisations ». Ce projet étant construit en deux phases indépendantes répondant chacune à un besoin particulier, ce rapport les couvrira de la même manière, en deux parties distinctes.

La maîtrise d'ouvrage est représentée par Elena PIERAZZO, enseignante-chercheuse en humanité numérique au Centre d'Etudes Supérieures de la Renaissance. La maîtrise d'œuvre est représentée par Olivier MILLOCHAU, étudiant en 5ème année à Polytech Tours, département informatique, encadré par Mohamed SLIMANE enseignant à Polytech Tours, département informatique, et Marie-Gabrielle LAGASSE, étudiante en 5ème année en humanité numérique au Centre d'Etudes Supérieures de la Renaissance, auteurs de ce cahier des spécifications.

1 Enjeux et Contexte

Le projet BONHum, BOccace Numérique Humaniste, est une Édition numérique des œuvres de Boccace qui consiste à étudier la prosopographie et la sémantique au sein des textes. Ce travail a pour but de faire ressortir l'essence humaniste qui jalonne les écrits de l'auteur et vise à mettre en lumière le rôle fondamental de Boccace (1313-1375) pour la diffusion de l'Humanisme dans l'espace culturel européen [2].

Les nombreux ouvrages de Boccace sont bien le véhicule privilégié de la nouvelle sensibilité humaniste en Europe. Ainsi, pour exposer avec clarté le caractère humaniste des écrits de Boccace, une collaboration scientifique internationale, des outils d'analyse et de valorisation, ainsi que la création d'une archive incrémentale de textes ont été mis en place. Ce projet étant un travail de recherche, il est destiné à s'élargir en termes de textes, images et partenaires [2].

Giovanni Boccaccio est né en 1313 à Florence où il y meurt en 1375. Il est considéré avec Dante et Pétrarque comme étant le fondateur de la tradition littéraire italienne et de la culture humaniste qui seront des sources d'inspiration pour la Renaissance européenne [1]. Boccace est issue de la bourgeoisie intellectuelle, une nouvelle classe sociale qui se développe à cette époque. Il écrit ses œuvres de jeunesse en italien vernaculaire et la postérité considère son recueil de nouvelles, Le Décaméron, qui connu un grand succès, comme l'incubateur de la littérature italienne en prose et le symbole de la liberté des mœurs. Par la suite, dans la plupart de ses œuvres, Boccace adoptera un caractère plus doctrinal et écrira en latin [1].

Actuellement, parmi les œuvres de Boccace, le travail d'édition numérique a été effectué pour :

- 16 textes du *Decameron*.
- 12 textes du *De mulieribus claris*.
- 9 textes de *De casibus vivorum illustrium*.
- 3 textes des traductions françaises de 1400 et 1409 du *De casibus vivorum illustrium* dont la traduction française est *Des cas des nobles hommes et femmes*.

À cela s'ajoute :

- Un travail de balisage en XML des textes.
- Les annotations d'images dans une base de données.
- Les données prosopographiques (une partie en base de données et une partie en XML) accessibles via un système de filtrage.

2 Objectifs

L'objectif de cette collaboration est de développer des outils pour l'exploitation de la prosopographie dynamique. Pour ce faire, les deux objectifs principaux sont les suivants :

- Créer une ontologie permettant de regrouper la base de données relationnelles (BDR) et les fichiers XML des textes de Boccace présents sur le site de BonHum afin de créer un modèle de données qui soit formel (modèle RDF/OWL).
Ce modèle de données sera ensuite destiné à être mis en ligne pour être exploitable par le plus grand nombre au travers d'une API Linked Open Data.
- Réfléchir à des outils de visualisation des données prosopographiques pour représenter l'évolution socio-professionnelle et psychologique du personnage en fonction de la temporalité du récit.

Concernant le second point, toute la difficulté se trouve dans l'idée de lier ensemble les mutations, les évolutions, les changements que peut avoir un même personnage au cours du temps. En effet, un même personnage accède dans son développement à certaines caractéristiques qui lui sont propres et qui sont souvent dues aux rencontres qu'il fait, aux occasions qui se présentent ou pas à lui et qui impactent sa vie au cours du récit. La complexité de cette représentation se situe par les occurrences de ces rencontres et de ces états. En effet, le personnage peut être amené à rencontrer à plusieurs reprises un autre personnage, mais sur une échelle de temps de différentes et avec un statut social et une fonction différente. De même au cours de sa vie, il peut changer constamment d'état passant ainsi d'heureux à malheureux et de malheureux à heureux. Il faut trouver une manière de représenter ces évolutions dans le temps de manière fluctuante et de ne pas tomber dans l'invariant. On cherche à donner de la profondeur et de l'épaisseur et à montrer des niveaux d'évolution en fonction des couches chronologiques.

Ce projet se constitue donc en deux phases, premièrement, la réalisation d'un tout nouveau système d'informations sous la forme d'une API, fonctionnant indépendamment de l'application Bonhum et deuxièmement, la mise au point de divers outils de visualisations de données fournis par cette précédente API.

3 Organisation

Comme énoncé plus haut, ce PRD est le fruit de l'association entre Polytech et le Centre d'études supérieures de la Renaissance, entre deux étudiants de ces écoles : moi même pour Polytech et Marie-Gabrielle Lagasse pour le CESR. De par cette situation particulière, nous avons travaillé en forte proximité l'un de l'autre pendant toute cette première partie de PRD.

D'un point de vue organisationnel, nous avons respecté les pratiques agiles : le projet étant dirigé

par Mme Elena Pierazzo, nous nous retrouvions chaque semaine lors d'une réunion faisant office de sprint où nous présentions les avancées récentes et les problèmes rencontrés et planifions les prochaines tâches à réaliser.

En dehors de ces réunions, une communication constante était assurée au sein de l'équipe et nous avons su allier nos domaines d'expertises différents. L'analyse des besoins et le cadrage du projet ont été réalisés par nous deux tandis que je me suis occupé de la modélisation et de la rédaction des spécifications techniques et Marie-Gabrielle de la mise au point des écrans et de leurs maquettes associées.

Nous avons aussi pu solliciter l'aide, les conseils ou l'expertise de nombreuses personnes au cours de ces quelques mois lorsque le besoin se faisait sentir ou que nous devions surmonter certains points de blocage.

2

Description générale

1 API

1.1 Environnement du projet

L'API ayant pour objectif de fonctionner indépendamment de l'application Bonhum, elle ne s'intégrera pas dans son environnement et en disposera d'un qui lui est propre.

On retrouve dans cet environnement une ontologie mise au point par Marie-Gabrielle. C'est cette ontologie qui servira d'entrepôt de données aux données des objets « Personnages », déjà présent dans l'application Bonhum et qui fournira ces dernières au format RDF à l'API.

L'API sera développée avec Python comme langage de programmation associé au framework Django. Seront aussi utilisés le framework Django Rest Framework, outil très puissant pour créer des API web et la librairie sparql-client pour générer des requêtes SparQL et communiquer avec l'ontologie. Tous ces outils seront dans leur dernière version à l'heure où est rédigé ce document. Les développements se feront dans l'éditeur de texte Visual Studio Code.

Une fois l'API terminée, elle sera vouée à être hébergée sur un serveur de l'université.

1.2 Caractéristiques des utilisateurs

De par sa nature, l'API est avant tout destinée à être utilisée par des développeurs.

Par contre, les données qu'elle fournira sont elles à l'attention de chercheurs en humanité numérique, d'où l'importance du respect du format de données RDF, il est nécessaire que les données soient inter-opérables.

Enfin, il n'y a aucun droit d'accès à l'API dû à son standard « Linked Open Data »

1.3 Fonctionnalités du système

En tant qu'API Web Linked Open Data, son unique fonction sera de fournir les données des personnages des œuvres de Boccace par le biais de requêtes HTTP.

Deux requêtes seront possibles :

- Une qui fournira les données d'un personnage en particulier.

- Une autre qui fournira les données de tous les personnages, avec possibilité d'y ajouter divers filtres.

1.4 Structure générale du système

L'architecture de notre API correspondra à celle d'une API Django Rest Framework classique. On y retrouvera 2 blocs majeurs :

- Le « ViewSet » : Ils sont responsables du traitement de demandes spécifiques, on peut les retrouver sous le nom de « Controller » dans d'autres framework. Dans notre API, on retrouvera un unique ViewSet (dédié à l'objet « Personnage ») qui comportera deux méthodes, chacune dédiée à une des deux requêtes présentées plus haut.
- Le « Router » : Détermine à quel « ViewSet » transmettre une requête entrante. A cette architecture s'ajoute l'ontologie RDF-OWL mise au point par Marie-Gabrielle, source de données au format RDF des entités « Personnages » dont l'API servira de point d'ouverture sur le web.

Le système entier embarquera lui l'ontologie en plus de l'API décrit plus haut.

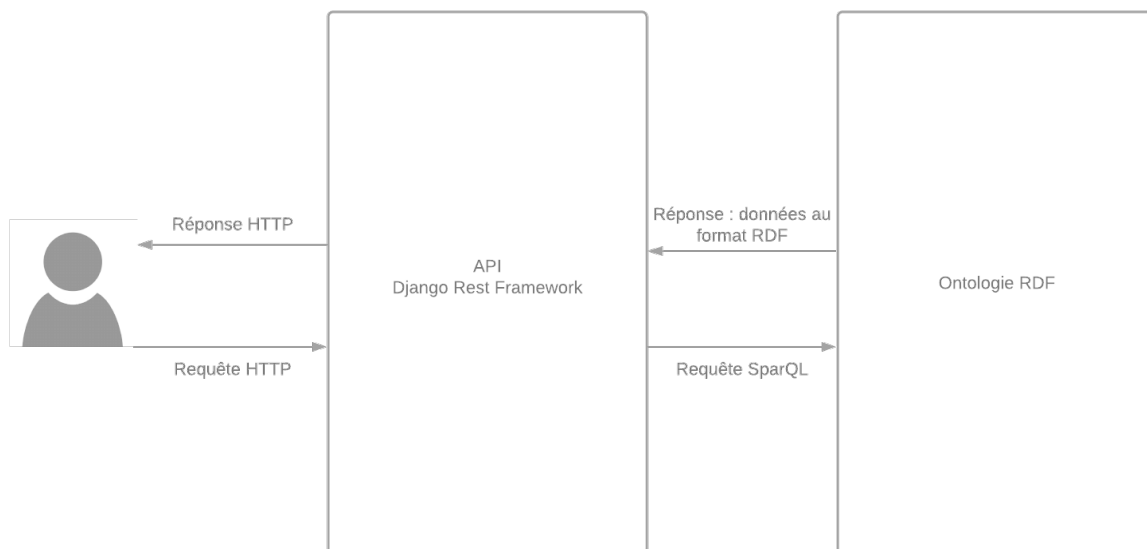


Figure 2.1 – Schéma de l'architecture et du fonctionnement général du système

2 Outils de visualisation

2.1 Environnement du projet

Ces outils de visualisation s'intègrent au sein de l'application Bonhum.

L'environnement du projet va donc dépendre de celui de l'application, c'est-à-dire que les développements se font en python version 2.7 sur le framework Django en version 1.8 et en Javascript. Nous utiliserons en plus la librairie D3.js pour créer les visualisations mêmes. L'éditeur de texte Visual Studio Code sera utilisé pour développer.

Les outils visent à représenter l'évolution d'objets « Personnage » et de leurs composantes, dans un premier temps, leur statut socio-économique et leur état psychologique. Les données de personnages sont contenues dans une base de données PostgreSQL et dans des listes XML. La manipulation de ces dernières se fera au travers de requêtes déjà implémentées dans l'application.

2.2 Caractéristiques des utilisateurs

Ces outils sont destinés à une utilisation régulière par des chercheurs en humanité numérique qui possèdent déjà une certaine connaissance de l'application existante.

Aucun droit d'accès particulier ne contraigne l'utilisation des outils de visualisation, tous les utilisateurs pouvant accéder à l'application peuvent accéder aux outils.

2.3 Fonctionnalités du système

L'objectif est d'afficher une représentation visuelle de l'évolution d'une composante d'un objet « Personnage » au cours du temps pour un ou plusieurs textes donnés (à des fins de comparaisons de textes).

Lorsque l'utilisateur est sur la page d'un personnage, il a la possibilité d'accéder à un onglet « Visualisation ». Une fois dans cet onglet, il est possible de sélectionner une composante ainsi qu'un ou plusieurs textes, parmi tous ceux où apparaît le personnage. Une fois la sélection réalisée, un graphe s'affiche. Il est possible sur certaines représentations, d'avancer ou de reculer dans le temps grâce à un curseur défilant.

2.4 Structure générale du système

La portée du système va se limiter à deux fichiers.

Le premier, « record_characters.html », est la page HTML même de consultation d'un personnage. Elle sera amenée à être modifiée pour accueillir un nouvel onglet « Visualizations » qui contiendra notre système de visualisations de personnages.

On retrouvera dans cet onglet la possibilité de sélectionner un ou plusieurs textes parmi tous ceux liés au personnage ainsi qu'une composante à visualiser (dans un premier temps, les relations sociales et l'état psychologique »).

Sera ensuite affiché un graphe représentant l'évolution de la composante choisie au sein des textes choisies.

Le second fichier sur lequel nous serons amenés à travailler, « visualizations.js », contiendra les méthodes Javascript permettant d'afficher les graphes grâce à la librairie D3.js.

3

État de l'art / Veille technologique

Afin d'avoir une idée assez précise des « canons » de design pour la visualisation des données prosopographiques, il convient d'analyser quelques interfaces inscrites dans l'univers de l'analyse textuelle et la modélisation d'outils visuels interactifs. Afin de découvrir ces interfaces nous avons majoritairement utilisé nos recherches sur le web et lorsque cela était possible, nous avons testé leur mise en application grâce à nos machines respectives.

Le domaine de la recherche se prémunit d'outils numériques pour étudier, préserver et partager le savoir et la connaissance. Les recherches en sciences humaines et sociales utilisent le traitement informatique des données et superposent à leurs travaux une coloration numérique. Cet entrecroisement entre les Humanités et le numérique, les Digital Humanities, a vu émerger de nouveaux outils dans l'enseignement, la recherche et la médiation, de nouvelles méthodes de travail et surtout, de nouvelles expériences. Aujourd'hui, un grand nombre de dispositifs numériques sont déployés au cœur des sujets d'études et dans l'enseignement supérieur : OCR, logiciel de textométrie, logiciel de gestion, logiciel d'intégration de données, logiciel de modélisation, etc. Les instituts, les organismes de recherche et d'enseignement se sont très vite emparés des avancées technologiques dans le but de promouvoir leurs théories et leurs observations. Ainsi, ils ont pu valoriser et rendre accessible les connaissances issues de leurs expériences.

À la suite de la multiplicité des supports de visualisations disponibles il faut prendre en compte toute une série d'adaptations. Il faut privilégier une visualisation graphique avec une ergonomie responsive. L'utilisateur doit être accompagné pas à pas dans sa recherche à travers un chemin indiqué de manière évidente et des interactions simples. C'est pourquoi il faut être vigilant à l'organisation et à la hiérarchisation des contenus.

1 Write.control

Write.control une plateforme d'écriture pour les auteurs développée par Maxime Dhalluin avec l'aide du programme « Start by Euratech » de l'incubateur de start-ups EuraTechnologies. Le projet compte à présent près de 8 000 utilisateurs. C'est une interface gratuite, disponible en ligne et sécurisée. La plateforme est capable de fonctionner sans connexion internet, en mode offline.

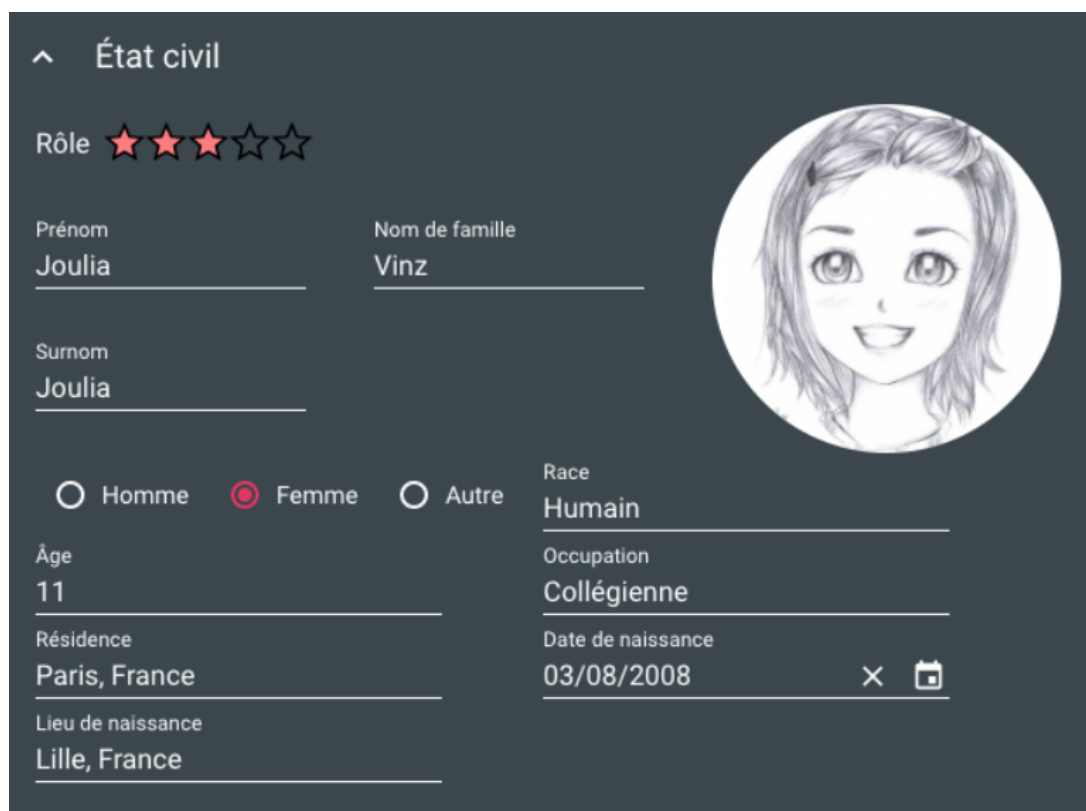
C'est un outil d'écriture créative proposant diverses fonctionnalités pour écrire un livre. Ainsi on retrouve des options d'écriture et d'organisation, des outils linguistiques, des fonctions concernant l'environnement de travail, le stockage et la sauvegarde des données ainsi que l'export pour le

changement de format. Concernant l'écriture et l'organisation de l'écriture, il y a la possibilité de créer des fiches personnages avancées. C'est cette dernière fonction qui va particulièrement nous intéresser.

L'option « Fiches de personnages » permet de créer pour chaque personnage une fiche très détaillée reprenant différentes caractéristiques tel que l'importance du rôle, l'état civil, l'âge etc. mais aussi la description physique et psychologique du personnage. Ces dernières sont représentées par un graphique en étoile modulable qui expose l'ensemble de ces données.

État civil

Cette partie permet de renseigner la carte d'identité de notre personnage. Ainsi on peut lui donner un prénom/nom/surnom, une date de naissance, un âge, un lieu de naissance et de vie, un sexe, ainsi qu'une photo de profil. On peut également renseigner son occupation/travail et le rôle qu'il tient dans le roman.



^ État civil

Rôle ★★★★★

Prénom
Joulia

Nom de famille
Vinz

Surnom
Joulia

☐ Homme ☒ Femme ☐ Autre

Race
Humain

Âge
11

Occupation
Collégienne

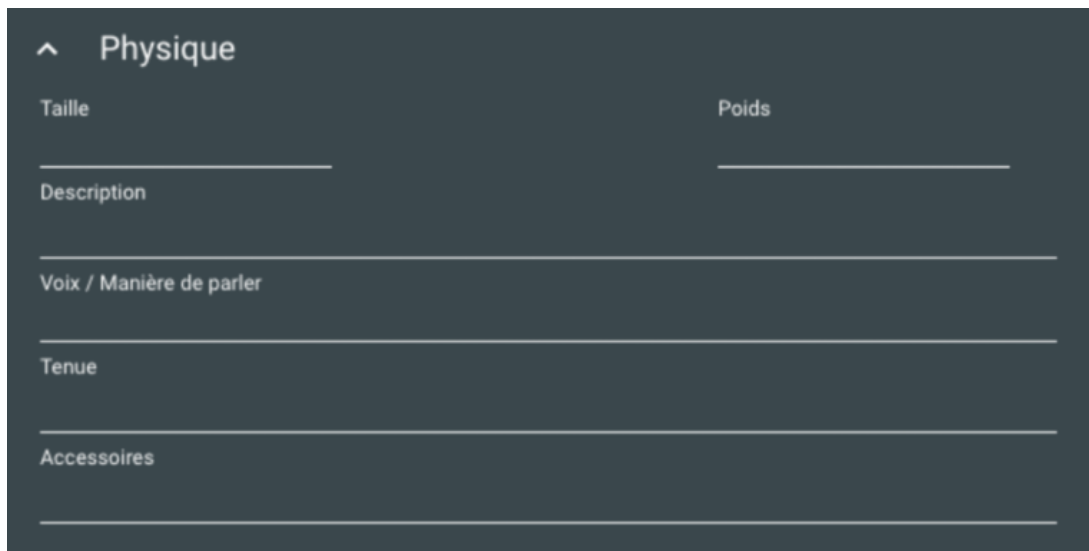
Résidence
Paris, France

Date de naissance
03/08/2008

Lieu de naissance
Lille, France

Physique

Cette partie permet de détailler les traits de notre personnage et notamment sa description physique.



The image shows a dark-themed user interface for a character creation form. At the top, there is a header with an upward-pointing chevron icon and the title 'Physique'. Below the header, the form is organized into several sections, each with a label and a corresponding input field represented by a horizontal line. The sections are: 'Taille' (Height) and 'Poids' (Weight) at the top; 'Description' below them; 'Voix / Manière de parler' (Voice / Way of speaking) below that; 'Tenue' (Clothing) below that; and 'Accessoires' (Accessories) at the bottom. Each label is positioned to the left of its respective input line.

Caractère

L'onglet caractère permet de donner à notre personnage un éventail de qualités et de défauts et les tapant soi-même ou en choisissant à travers une liste déroulante une sélection proposée. Un graphe de personnalité modulable est alors proposé. L'utilisateur peut alors grâce à un bouton « + » au centre du graphe ajouter un élément et définir pour chaque élément un nombre de points allant de 0 à 20.

^
Caractère

Goût

Qualités caractérielles

Sérieux
Curieux
Ajouter une qualité...

Défauts caractériels

Sang-chaud
Sale
Ajouter un défaut...

Tics, manies, habitudes et addictions

Peurs et doutes

Intelligence [6]

[13] Robustesse

[9] Empathie

[13] Perception

Agilité [13]

Sociabilité [14]


Force [11]

Profil

Cette option permet de renseigner le profil culturel et social de notre personnage.

^ Profil

Éducation

Richesses 

Croyances et idéologies

Lieux marquants

Secrets

Phrases ou expressions typiques

Evolution

Enfin, l'onglet évolution permet de faire état de l'évolution de notre personnage au cours du récit avec un découpage tridimensionnel passé/présent/futur.

^ Évolution

Buts / Objectifs

Passé

Décrire le personnage avant le début de l'histoire (enfance, souffrances potentielles...).

Présent

Décrire le personnage au commencement de l'histoire.

Futur

Décrire le personnage à la fin de l'histoire.






















1.1 Analyse critique

Write.control est un éditeur de textes simple d'utilisation qui accompagne l'utilisateur au fur et à mesure de son récit pour l'aider à structurer sa pensée. Cet outil offre un cadre agréable et hiérarchisé qui répond aux besoins des auteurs : correction des fautes, ressources, détection des chapitres, dico, etc.). Le panorama de fonctionnalités qu'il propose permet d'atteindre ses objectifs d'écriture et apporte des outils de gestion et d'organisation bénéfiques à la création. Concernant la création des personnages avec des fiches avancées, la représentation en étoile du caractère des protagonistes est visuellement pertinente et permet d'avoir une approche directe et claire préférable à une simple liste d'adjectif. Il serait donc intéressant de faire de même pour représenter graphiquement l'espace-temps et les interactions qu'un personnage peut avoir avec d'autres à la façon d'une frise chronologique faisant état des différents événements et des acteurs importants. En effet, l'onglet « évolution » pourrait se munir d'une option graphique pour l'aspect visuel et garder son bloc « Passé/Présent/Futur » pour l'aspect purement écriture.

2 Voyant

Voyant Tools est outil d'analyse automatique de texte ou text mining. C'est une application Web open source permettant d'effectuer une analyse poussée des textes ou des corpus de texte. Ce projet open-source est dirigé par les universitaires canadiens Stéfán Sinclair et Geoffrey Rockwell et le code est disponible sur GitHub.

Voyant Tools peut accepter une grande variété de formats, y compris le texte brut, HTML, XML, PDF, RTF, et MS Word. Il offre un large éventail de visualisations de données en fonction de nos besoins. En effet, il contient 21 outils pour représenter et analyser un corpus.

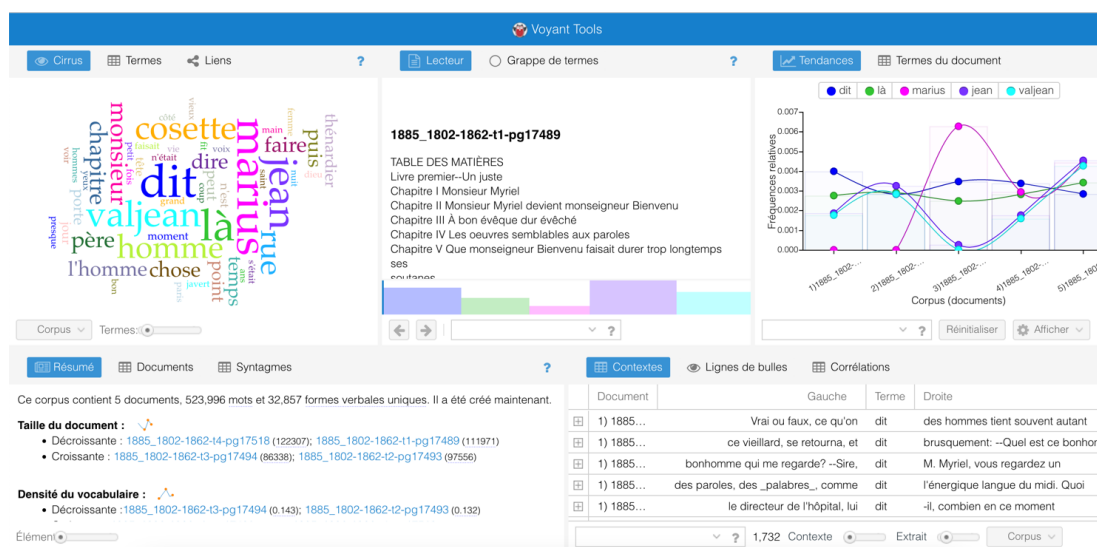
	Bubblelines visualizes the frequency and repetition of a word's use in a corpus. Each document in the corpus is represented as a horizontal line and divided into segments of equal length.		Mandala allows the importing of "textual" files to perform analysis on the frequency and linkage of terms.
	Bubbles represents the relative frequency of words in a corpus through a cloud of bubbles.		Reader provides a viewing window to allow the user to read the full text of the corpus that they have imported into Voyant Tools.
	Cirrus is a word cloud displaying the frequency of words appearing in a corpus.		RezoViz visualizes the relationships between people, locations and organizations in a collection of documents.
	Corpus Grid provides an overview of a corpus, displaying each document's title, total number of words (word tokens), number of unique words (word types), and lexical density (the ratio of tokens to types).		ScatterPlot displays the correspondence of word use in a corpus.
	Corpus Summary is a tool that provides a simple, textual overview of the current corpus.		Termometer depicts the change of the frequency of word across a corpus spread over time.
	Corpus Type Frequencies Grid provides an ordered list for all the words' frequencies appearing in a corpus.		TermsRadio provides a scrolling line graph that can depict the change of the frequency of word across a corpus spread over time.
	Document Input Add allows the user to dynamically add additional documents to a corpus after importing the initial texts.		Type Frequencies Chart shows a line graph depicting the distribution of a word's occurrence across a corpus.
	Document Type Collocate Frequencies Grid provides an ordered list of word collocation for a specified word and document.		Word Count Fountain visualizes word frequencies as a fountain.
	Document Type Frequencies Grid provides an ordered list of word frequencies along with other statistical data for each document in a corpus.		Links represents the collocation of terms in a corpus by depicting them in a network through the use of a force directed graph.
	Document Type KWICs Grid displays a table contextualizing a selected word with the phrases or paragraphs of text that directly precede and follow each instances of the word throughout the corpus.		Lava displays multiple levels of a corpus in a three-dimensional environment.
	Knots represents a corpus as a series of twisted lines.		

Cette application permet d'avoir accès à une palette de contrôle où l'on peut choisir les outils que l'on souhaite appliquer sur nos textes. Ils sont simples d'utilisation et visuellement pertinents. La page d'accueil nous permet d'ajouter des contenus à analyser. En effet, il est possible d'ajouter :

- Des URL de texte en ligne, y compris des URL multiples en changeant de lignes.

- Un texte, par copier-coller.
- Un corpus de documents présents dans nos documents personnels qui sera chargé sur la plateforme.

Marie-Gabrielle a choisi pour cette démonstration de mise en pratique un corpus de texte des Misérables de Victor Hugo et de présenter uniquement les outils qui se présentent à nous de manière standard, c'est-à-dire : Cirrus, Lecteur, Tendances, Résumé et Concordances.



Nous avons une page de résultat qui se découpe en cinq blocs. En partant de la gauche vers la droite et de haut en bas, on trouve par défaut les outils suivants :

Cirrus : un nuage de mot

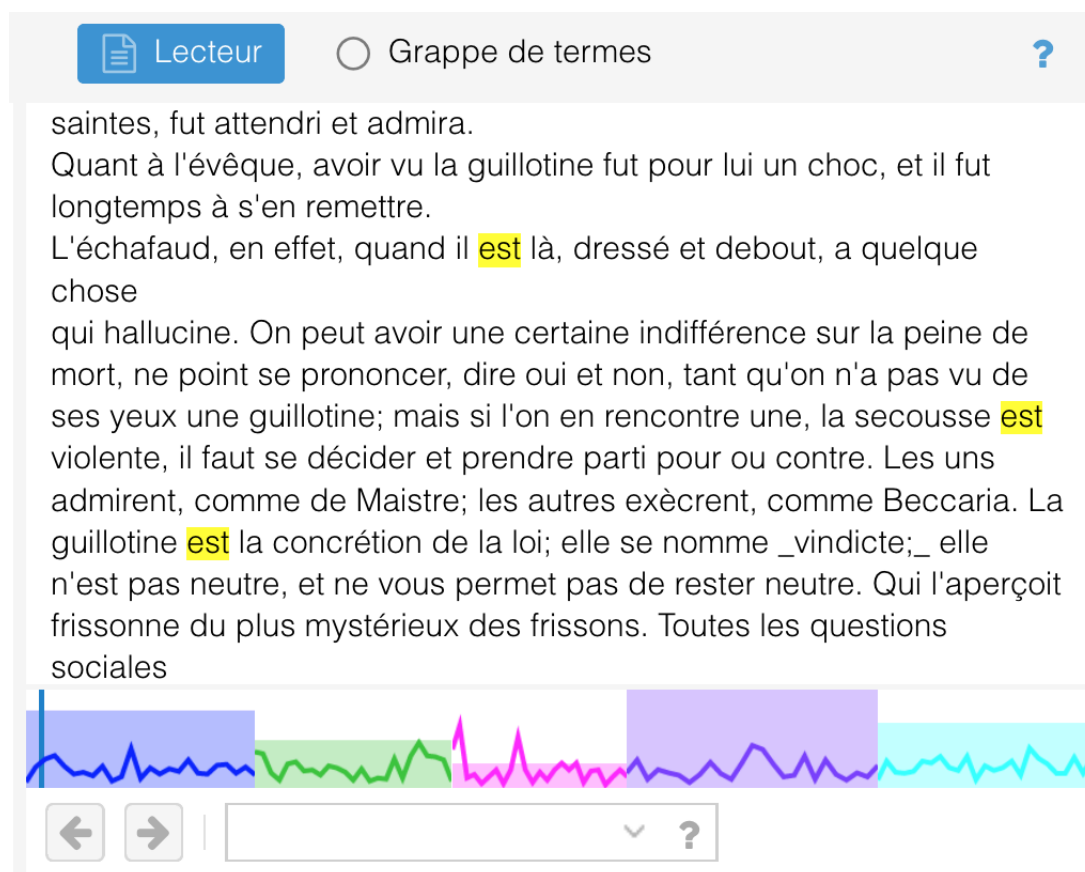
Nous avons une imbrication de mots ou la taille de la police est égale à leur fréquence de citation dans le corpus. On peut choisir grâce à un onglet se trouvant en bas de la fenêtre, le corpus entier ou bien un texte en particulier. Il est également possible grâce à un curseur scale (à côté droite de l'onglet « Corpus ») d'afficher le volume de mots dont on veut voir la fréquence d'apparition.



Il est également possible de retirer des mots clés indésirables qui viennent polluer le graphique et qui ne sont pas pertinents pour l'analyse. Pour les faire disparaître, il suffit de cliquer sur « Définir les options de cet outil ». Puis de cliquer sur « modifier la liste » et de rajouter à la liste de mots déjà préexistant le mot que vous désirez ne plus voir.

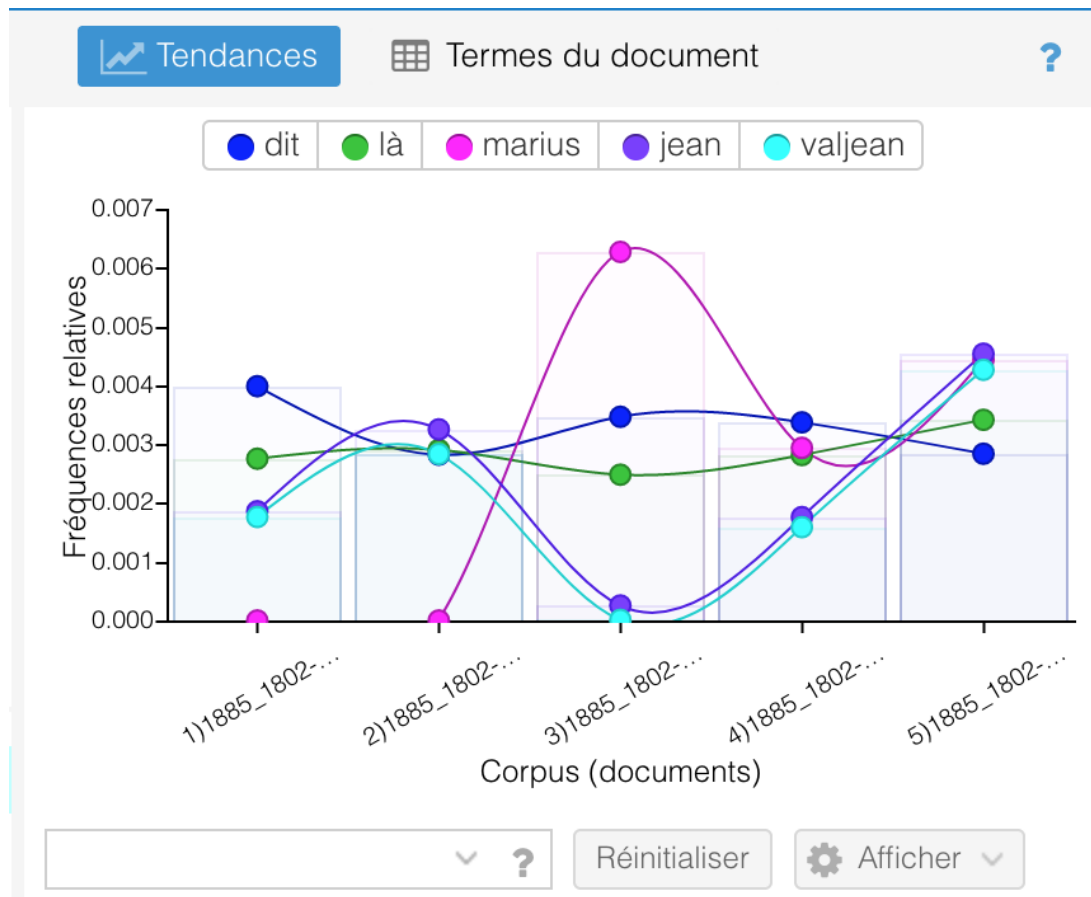
Lecteur : Le corpus de texte

Il nous permet d'accéder au texte brut. On peut y naviguer en se déplaçant de haut en bas avec la molette de notre souris ou bien grâce au petit histogramme de fréquence accessible en dessous du texte. On peut également si l'on clique sur un mot du texte qui devient alors sur surligné en jaune, voir l'endroit où il se situe sur l'histogramme et l'emplacement de ses occurrences dans les autres textes du corpus.

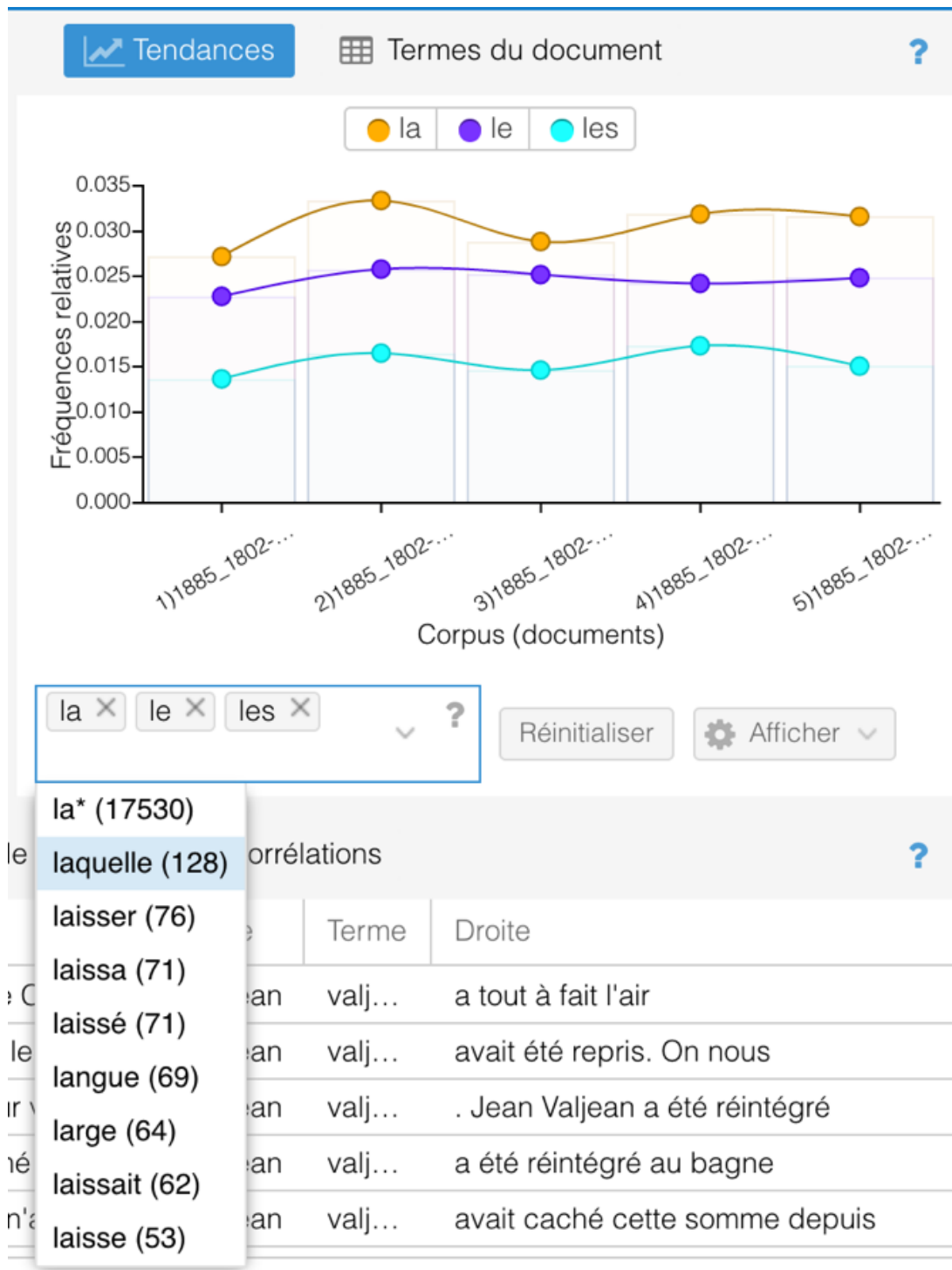


Tendances

De façon standard, cet outil représente graphiquement les courbes de fréquences d'apparition des mots-clés les plus présents dans chaque texte du corpus. On peut en glissant avec notre souris sur un des points de couleur voir apparaître le mot dont il est question, sa fréquence et à quel texte il appartient dans le corpus.



À l'aide de la barre de recherche en-dessous du graphique on peut écrire un mot ou plusieurs dont on souhaite voir la fréquence. Il est également possible de sélectionner directement dans une liste déroulante des mots proposés par le moniteur commençant par la même syllabe que le dernier mot choisi ou écrit.



Résumés

Le nombre de documents analysés (ici 5 documents), de mots dans les documents et les mots clés les plus présents. Puis en déroulant cet onglet, on peut lire les informations concernant la taille et la densité du vocabulaire, la moyenne des mots par phrase et les mots distinctifs.

Résumé Documents Syntagmes ?

Ce corpus contient 5 documents, 523,996 mots et 32,857 formes verbales uniques. Il a été créé maintenant.

Taille du document :

- Décroissante : 1885_1802-1862-t4-pg17518 (122307); 1885_1802-1862-t1-pg17489 (111971)
- Croissante : 1885_1802-1862-t3-pg17494 (86338); 1885_1802-1862-t2-pg17493 (97556)

Densité du vocabulaire :

- Décroissante : 1885_1802-1862-t3-pg17494 (0.143); 1885_1802-1862-t2-pg17493 (0.132)
- Croissante : 1885_1802-1862-t1-pg17489 (0.117); 1885_1802-1862-t4-pg17518 (0.124)

Moyenne des mots par phrase :

Élément

Contextes

Il nous permet de trouver ici le mot sélectionné dans les phrases où il est utilisé avec le rappel du document auquel il est rattaché dans le corpus ainsi que sa position dans le document.

Contextes Lignes de bulles Corrélations ?

Document ↑	Gauche	Terme	Droite	Position
2) 1885_1...	Innocente Chapitre I...	valj...	a tout à fait l'air	564
2) 1885_1...	devient le numéro 9...	valj...	avait été repris. On ...	21655
2) 1885_1...	pour vol, et nommé ...	valj...	. Jean Valjean a été ...	21819
2) 1885_1...	et nommé Jean Valj...	valj...	a été réintégré au b...	21821

valjean x ? 276 Contexte Extrait Corpus

2.1 Analyse critique

Voyant est un outil complet et très abouti aux multiples fonctionnalités pour l'analyse de texte. Sa division en différents blocs ainsi que la présentation des différents outils aux formats dynamiques et graphiques offrent une représentation visuelle riche et bien pensée.

Cependant la puissance de l'outil pourrait déformer l'œuvre selon le parti de l'utilisateur d'utiliser un outil plutôt qu'un autre. De plus, il dispense la lecture complète de l'œuvre, ce qui peut être vu comme un gain de temps intéressant mais qui ne peut se substituer à un expert du sujet qui aura une expérience et une approche plus sensible sur les textes. On peut également souligner le fait que la quantité des fonctionnalités réunies sur une même fenêtre donne un caractère massif et impressionnant à l'outil et qu'un non initié peut rapidement se perdre dans son abondance de possibilités d'analyse. Par ailleurs, il n'est pas toujours possible pour certains outils de revenir à l'état initial une fois que l'on a appliqué certains filtres, excepté pour l'outil « Tendances » qui propose un bouton « Réinitialiser ». En effet, pour l'outil « Liens », il est nécessaire de cliquer sur un autre outil, puis de retourner sur « Liens » pour retourner au graphique d'origine. De plus, si l'on a choisi de charger un corpus de textes, tous les outils ne permettent pas de choisir si l'on souhaite définir leur application sur le corpus entier ou sur un document en particulier comme peut le faire « Cirrus » par exemple. Il est alors nécessaire de revenir à la page d'accueil et de charger un texte seulement.

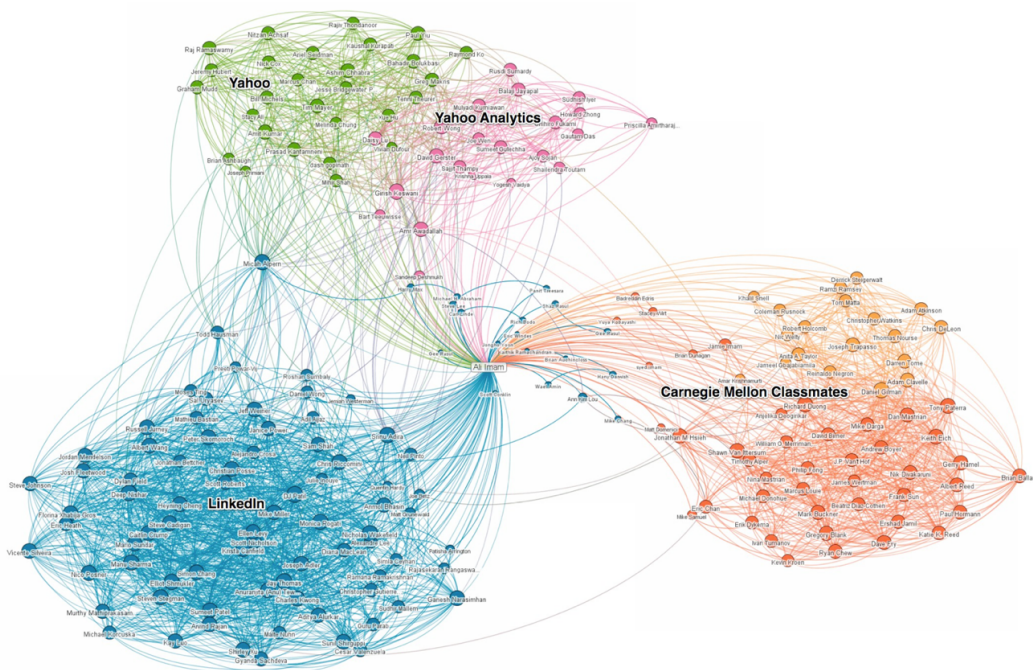
3 LinkedIn Maps

LinkedIn Maps est une fonctionnalité du réseau social professionnel en ligne LinkedIn Corporation créé en 2002. LinkedIn développe en 2011 un nouvel outil dans son Linked Labs, appelé InMaps, permettant de créer une carte interactive afin de visualiser ses relations professionnelles. De plus, InMaps permet non seulement de visualiser graphiquement l'ensemble de ses connexions mais également de comprendre les relations établies entre nous-même et les personnes de notre réseau. Le principal scientifique des données sur LinkedIn, Imam Ali, explique comment cette représentation en réseau peut être utile :

"You can use those insights to measure your own impact or influence or create opportunities for someone else. So, you might see two distinct groups that you could introduce to become one. Or you might leverage one person to connect them to someone else. See an area that doesn't look like it is representative of your professional world? Fix it by adding the necessary connections. "

Afin de créer son graphe de réseau, il faut se rendre sur le portail <http://inmaps.linkedinlabs.com> et se connecter avec son nom d'utilisateur et son mot de passe sur LinkedIn. InMaps crée alors rapidement une carte montrant les connexions entre les gens de son réseau LinkedIn. Cependant, le portail dont il est question amène vers une page dont l'adresse est introuvable. Il est donc possible que l'outil qui est sorti il y a plus de 10 ans ne soit plus disponible aujourd'hui. Cependant, si on reprend les différents articles et les essais de mise en pratique de 2010 et de 2011, on peut se rendre compte du potentiel de l'outil.

En effet, si votre réseau était composé d'au moins 50 contacts, vous pouviez créer votre graphe. Pour notre exemple nous prenons la map de Imam Ali.



On peut voir ci-dessus, que les contacts sont séparés en différents clusters ou groupes (ici 3 groupes distincts). Chaque couleur représente l'appartenance à un réseau professionnel. Une légende permet de changer le nom de l'étiquette pour identifier le facteur commun de chaque groupe de couleur.

Si vous passez votre curseur sur un contact d'un même groupe de couleur, une petite fiche d'identité apparaît et des liens se créent reliant le contact sélectionné à d'autres contacts comme une toile d'araignée. Ainsi la figuration des relations de chaque personne est représentée. Par ailleurs, on observe que des relations partagent des airs de connexions étendues entre nos différents groupes professionnels.



3.1 Analyse critique

Cette représentation de carte relationnelle est pertinente car elle permet d'aider l'utilisateur à faire état de son univers professionnel mais également de se rendre compte des lacunes dans ses connexions. Il nous permet ainsi d'effectuer un bilan socio-professionnel et de mettre en évidence des secteurs sur lesquels être plus actif afin d'exploiter au mieux son réseau et développer son influence.

Cependant, il est nécessaire que nos contacts aient un compte sur LinkedIn pour être identifiés par la base de données de ce dernier. Il est donc indispensable d'utiliser d'autres réseaux sociaux pour avoir une vue d'ensemble complète. De plus, pour pouvoir créer sa carte il est nécessaire d'avoir au moins 50 contacts, ce qui exclut les utilisateurs ayant des relations peu étendues alors que l'outil prêtant justement aider à combler ses lacunes dans sa sphère professionnelle. Pour finir, il serait intéressant que cet outil propose une nouvelle mise à jour de la version de InMaps afin qu'elle soit toujours disponible sur le Web.

4

Analyse et conception

1 API

1.1 Analyse

L'API se présente comme une application fonctionnant indépendamment de l'application Bonhum déjà existante, c'est donc un nouveau projet qui a été étudié et analysé comme tel. Ainsi nous avons pu, au cours d'une première réunion avec Mme Elena Pierazzo, cadrer les besoins : rendre les données de l'application Bonhum interopérables et facilement accessibles et moissonable sur le web tout en satisfaisant les standards du web sémantique au travers d'une API Linked Open Data.

Le Linked Open Data est un système reposant sur 4 principes, où les données sont :

- Libres de toute licence.
- Conforme au modèle de données "Resource Description Framework" (RDF).
- Identifiées de manière unique et permanente.
- Mise en ligne en suivant le protocole HTTP.

Afin de répondre à ces critères, il nous a paru évident qu'utiliser les outils du web sémantiques était nécessaire. Une première difficulté s'est présentée au travers du peu de connaissances que nous avions lié à ces outils. En effet, le web sémantique étant un sujet non enseigné à Polytech et que j'ai découvert avec ce PRD, Marie-Gabrielle se présentait comme seule expertise sur la question.

C'est pourquoi nous avons sollicité l'aide de Mme Perrine Thuringer, professeur au CESR. C'est au cours d'une réunion avec elle que nous avons établi que le système se basera sur une ontologie web au format RDF, mise au point par Marie-Gabrielle, qui fera office d'entrepôt de données. Cette ontologie fournira ses données à l'API grâce à un système de requête en SparQL, un langage de requête et protocole propre au web sémantique permettant de manipuler des données RDF. L'API en elle-même n'est donc là que pour ouvrir l'ontologie au web.

1.2 Spécifications

Les fonctionnalités de l'API n'ont pour qu'unique but de récupérer les données contenues dans l'ontologie.

1.2.1 Fonction 1 : getOneCharacter

Cette fonction a pour but de fournir les données d'un personnage en particulier, spécifié par son nom.

1.2.2 Fonction 2

Cette fonction a pour but de fournir les données de tous ou d'une partie des personnages, il sera possible d'appliquer des filtres à la requête pour préciser le résultat voulu.

2 Outils de visualisations

2.1 Analyse

Les outils de visualisations se sont très vite présentés comme la demande principale du sujet. Plusieurs réunions ont lieu avec Mme Elena Pierazzo et Mme Sabrina Ferrara, une autre chercheuse en humanité numérique, chef du projet Bonhum aux côtés d'Elena. Très rapidement, le besoin majeur du projet a été énoncé : avoir la possibilité de plus facilement étudier les données afin d'en ressortir des informations difficiles à extraire actuellement, comme par exemple la comparaison de l'impact de diverses traductions d'un même texte sur l'évolution d'un personnage, sa représentation. Pour ainsi dire : l'étude de la donnée dans son contexte.

Nous en avons donc traduit la problématique suivante : Comment et sous quelle forme mettre en relation ces différentes données de manière à faciliter leur étude ?

Quatre grands points en ressortent, en cohérence avec le besoin initial :

- Coupler les informations cohérentes entre elles, par exemple visualiser le réseau social d'un personnage en même temps que son statut social.
- Juxtaposer les jugements de l'auteur aux visualisations afin d'ajouter un contexte supplémentaire à la donnée.
- Visualiser plusieurs représentations en même temps afin de faciliter la comparaison de textes et de traductions.
- Et enfin, mettre en place des interfaces et des représentations graphiques intuitives et cohérentes.

Suite à cela, nous avons pu nous atteler à imaginer les premières maquettes, réalisées par la suite par Marie-Gabrielle grâce au logiciel Figma.

En parallèle, j'ai étudié les différentes options disponibles quant à la librairie de visualisation que j'allais utiliser pour réaliser les outils demandés. Le critère de recherche le plus important était la puissance de librairie, en effet, de par la diversité des données en notre possession, il était primordial d'avoir le plus de contrôle possible sur la manière de les représenter. Mon choix s'est porté sur la librairie D3.js, elle est plébiscitée dans le domaine de la visualisation de données sur le web et est la base de nombreuses autres librairies de visualisation grâce à sa puissance et aux fonctionnalités qu'elle offre.

J'ai ensuite contacté Mme Suzy Piat, l'ancienne développeuse de l'application Bonhum pour comprendre plus en détail le fonctionnement du code existant et planifier l'intégration des nouvelles fonctionnalités au sein du système : Les données d'un personnage sont actuellement stockées dans deux endroits différents, une partie correspondant aux informations "statique" (religion, sexe, traits de personnalités, etc ...) est stockée en base de données tandis qu'une autre partie, relative à toutes les caractéristiques d'un personnage au sein d'un texte (ses relations,

son état psychologique, etc ...) est stocké dans le texte même, via des annotations. Ainsi, lors du chargement de la page de consultation d'un personnage, tous les textes liés à un personnage seront récupérés et parcourus afin d'en extraire les informations relatives à ce dernier. Les outils de visualisations n'auront donc qu'à utiliser les données issues de cette reconstitution de caractéristiques.

C'est en étudiant ce système que j'ai relevé un problème majeur : il n'y avait actuellement aucun marqueur temporel sur les données, tout était "en vrac". Nous avions les informations sur l'évolution d'un personnage au sein d'un texte, mais pas le moment où elles ont eu lieu, il était donc impossible de les visualiser en l'état.

Après concertation avec Mme Pierazzo, il a été décidé d'ajouter une couche d'analyse syntaxique supplémentaire pour marquer nous-même les caractéristiques récupérées des textes. L'unité temporelle des données est donc un pourcentage du texte dans lesquelles elles se trouvaient.

2.1.1 Spécifications

Pour chaque fonctions des outils de visualisations, la génération d'une représentation graphique lui est associée.

2.1.2 Fonction 1 : drawRelationshipsNetwork

Cette fonction a pour objectif de dessiner un réseau présentant les relations sociales d'un personnage au cours d'un texte.

Chaque personnage sera représenté par un nœud étiqueté du nom du personnage.

Le personnage à visualiser apparaîtra au centre du réseau, il sera entouré et connecté à tous les personnages avec qui il partage un lien social à un moment donnée. Les arêtes seront étiquetées de la nature du lien social. Il sera possible de cliquer sur les nœuds pour afficher le graphe des relations sociales de ce personnage au sein du même texte. La dimension temporelle peut être manipulée grâce à un curseur défilant sous le graphe.

2.1.3 Fonction 2 : drawPsychologicalStatus

Cette fonction a pour objectif de dessiner un graphique présentant l'évolution du statut psychologique d'un personnage au cours d'un texte.

La représentation prendra la forme d'un graphique de différences par rapport à une valeur neutre. Toute valeur supérieure à cette valeur neutre à un moment donné dénotera du statut heureux pour le personnage et à l'inverse, une valeur inférieure dénotera du statut malheureux.

5

Mise en œuvre

Tous les développements se sont étalés sur 3 fichiers au total :

- “record_character.html” : Qui correspond à la page de consultation d’un personnage même.
- “search_character.py” : Qui contient différentes méthodes propres au backend de l’application.
- “visualizations.js”, qui contient toutes les méthodes d’affichages des graphes et de manipulation des données créées dans “search_character.py”.

Toute la mise en œuvre peut être séparée en 3 phases :

- Les développements propres aux backend même de l’application, exclusivement dans “search_character.py”.
- Les développements couvrant en même temps “search_character.py” et “visualizations.js” et portant sur la manipulation et la communication des données.
- Et enfin les derniers développements, uniquement frontend portant sur la création des graphes dans “visualizations.js”.

1 Déviation par rapport aux spécifications

Au moment des spécifications, ma binôme Marie-Gabrielle n’avait pas encore eu de cours sur les ontologies et moi-même découvrait encore le sujet au travers de ce projet. Il était donc compliqué de prévoir cette tâche. Finalement, l’ontologie développée par Marie-Gabrielle se suffit à elle-même et ne nécessite pas d’intervention de ma part. Le travail présenté dans la partie mise en œuvre de ce rapport ne porte donc que sur mes productions et ne couvrira pas l’ontologie.

2 1er développements

2.1 Analyse syntaxique

Comme expliqué dans la partie “Analyse et conception”, j’ai rapidement rencontré un premier point de blocage dans le fait que les données à visualiser ne portaient pas de marqueurs temporels, elles étaient pour ainsi dire “en vrac”. J’ai donc remonté le problème à Mme Pierazzo et après

étude de la question, il en est ressorti qu'il n'était pas possible qu'elle intervienne de son côté pour régler la situation. Il était donc nécessaire que je m'occupe de rajouter une couche de traitements à l'application pour marquer automatiquement les données.

Lorsque un texte est annoté, des balises sont créées autour du morceau de texte sélectionné pour l'annotation avec différents attributs permettant de caractériser cette dernière et d'indiquer, notamment, l'id du ou des personnages concernés par l'annotation. Ainsi, pour récupérer les caractéristiques d'un personnage, l'algorithme prend tous les textes liés au personnage et recherche les balises le concernant. La portée de cette tâche s'étendait donc à cet algorithme, que je devais modifier afin d'incorporer le marquage temporel des données. Plusieurs solutions ont été envisagées :

- La première a été de faire appel aux fonctionnalités de la librairie utilisée pour parcourir et extraire les données des textes : BeautifulSoup. BeautifulSoup est une librairie d'analyse syntaxique de documents HTML et XML et permet donc d'en extraire les éléments les composant.

Parmi les informations relatives aux éléments qu'il est possible d'y accéder, 2 m'ont paru prometteuses au premier abord : "sourceline" et "sourcepos", soit respectivement la position d'un élément dans son paragraphe et la position dudit paragraphe dans le texte.

Malheureusement tous les textes présents dans le projet bonhum ne sont pas structurés de la même manière, certains présente un unique grand paragraphe, appelant à utiliser l'attribut "sourceline", tandis que d'autres sont chapitrés en plusieurs paragraphes, propices à l'utilisation de l'attribut "sourcepos".

- La seconde a été de modifier un analyseur syntaxique existant, afin de l'adapter à nos besoins et de récupérer les annotations ainsi que leur position au sein des textes.

Bien que cette solution soit la plus élégante qui ait été envisagée, elle est aussi la plus gourmande en ressource et nécessite le plus de temps pour être implémentée.

- La troisième utilise la méthode ".find()" de python qui permet de trouver la position, si elle existe, d'une chaîne de caractères dans une autre. Dans notre cas, la chaîne de caractère à retrouver serait la valeur de l'annotation et la recherche serait effectuée sur le document dans sa totalité.

Cette solution, bien que fonctionnelle présente une certaine faiblesse : pour chaque annotation relevée, tout le texte est à nouveau parcouru, ce qui implique une certaine quantité d'opérations inutiles.

Malgré la faiblesse de la dernière solution, c'est tout de même celle-là qui a été retenue, car elle offre des résultats d'une précision convenable avec une certaine facilité d'implémentation.

Ainsi, il a été ajouté à l'algorithme de récupération des caractéristiques une étape supplémentaire : pour chaque annotation, une chaîne de caractères unique est ajoutée dans le texte qui lui est associé. Une recherche dans le document sur cette chaîne de caractères est ensuite effectuée puis est renvoyée la position obtenue divisée par la longueur du texte total fois cent. La valeur de renvoi est donc la position de l'annotation dans le texte, relative à la longueur de ce dernier, exprimé en pourcentage.

3 2nd développements

3.1 Communications des données

Passé le problème du marquage temporel des données, un autre obstacle a été rencontré marquant le début d'une seconde phase de développement, faisant intervenir pour la première fois le fichier "visualization.js". Il était, en effet, impossible d'utiliser directement les données issues de "search_character.py" dans "visualization.js".

Django fonctionne de la manière suivante : lorsqu'un utilisateur émet une requête, Django la traite à l'aide de différentes classes (correspondant donc à une partie du "backend") et renvoie la réponse (le plus généralement au "frontend" chargée de traiter les données). Nous avons dans notre cas le fichier "record_character.html" chargé d'afficher les données issues de "search_character.py". Si l'on veut utiliser ces données dans un fichier javascript associé il faut donc injecter ces dernières sérialisées au format JSON dans le script directement depuis le fichier "record_character.html". Là était le point le bloquant : les données telles qu'elles renvoyées par Django n'étaient pas sérialisables au format JSON. En effet, toutes les annotations et les caractéristiques étaient contenues dans un dictionnaire Python où certaines clés et valeurs étaient des objets propres au projet bonhum non sérialisable automatiquement. Il a fallu donc créer une méthode pour manuellement sérialiser le dictionnaire de données des annotations pour pouvoir l'envoyer au script javascript.

3.2 Ecran de sélection

À cette étape-là, les données étaient complètes (marqueurs temporels) et utilisables dans le script, il ne manquait donc plus qu'à créer l'écran de sélection utilisateur. Les styles ayant déjà été mis au point pour le reste de l'application, je n'ai eu qu'à modifier le fichier "record_character.html". J'y ai ajouté un nouvel onglet en suivant la structure existante comprenant deux groupes de boutons : le premier offre à l'utilisateur la possibilité de sélectionner n'importe quel texte associé au personnage et le second le graphe à afficher.

Character: Priamus

Information Texts (5) Characteristics Images (4) Visualizations

Sélectionner un ou plusieurs textes

De Priamus roy de Troie et Hecuba sa femme le XIIIe chapitre. (originel (edition))

Chapitre .XIII.e : De Priamus, roy de Troies, et de Hecuba, sa femme. Histoire abregee. (originel (edition))

Le XIIIe chapitre contient le cas de Priam roy de Troie, de Hecube sa femme, et de plusieurs autres nobles, commençant ou latin: "Origo preclarissima" (originel (edition))

De Priamo, Troianorum rege, et Hecuba (originel (edition))

De Hecuba regina Troianorum (originel (edition))

Sélectionner une caractéristique

Relations

Psychologie

Figure 5.1 – L'écran de sélection de textes et d'un graphe

Dès lors qu'au moins un graphe et un texte sont sélectionnés, une méthode sera appelée pour construire les données à afficher selon la sélection utilisateur à partir des données de toutes les annotations. Ainsi, l'utilisateur peut à loisir ajouter, supprimer des textes, et même changer de graphe et les données seront reconstruites dynamiquement. Si un utilisateur sélectionne un graphe pour lequel certains textes n'ont aucune annotation correspondante, alors ces textes seront automatiquement désélectionnés et désactivés tant que le graphe reste sélectionné.

4 3eme développements

Avec tous les problèmes réglés et l'interface utilisateur mises en place, il était enfin possible de se concentrer entièrement sur la réalisation des visualisations demandées.

4.1 Graphe de l'évolution psychologique des personnages

La première version du graphe fut assez simple et servit principalement de prise en main de la librairie D3.js. Elle comportait en tout :

- L'axe des abscisses, de 0 à 100 représentant un pourcentage d'un texte.
- L'axe des ordonnées, qualitatif entre bonheur et malheur.
- Une courbe où chaque point correspond à une annotation.

Character: Priamus

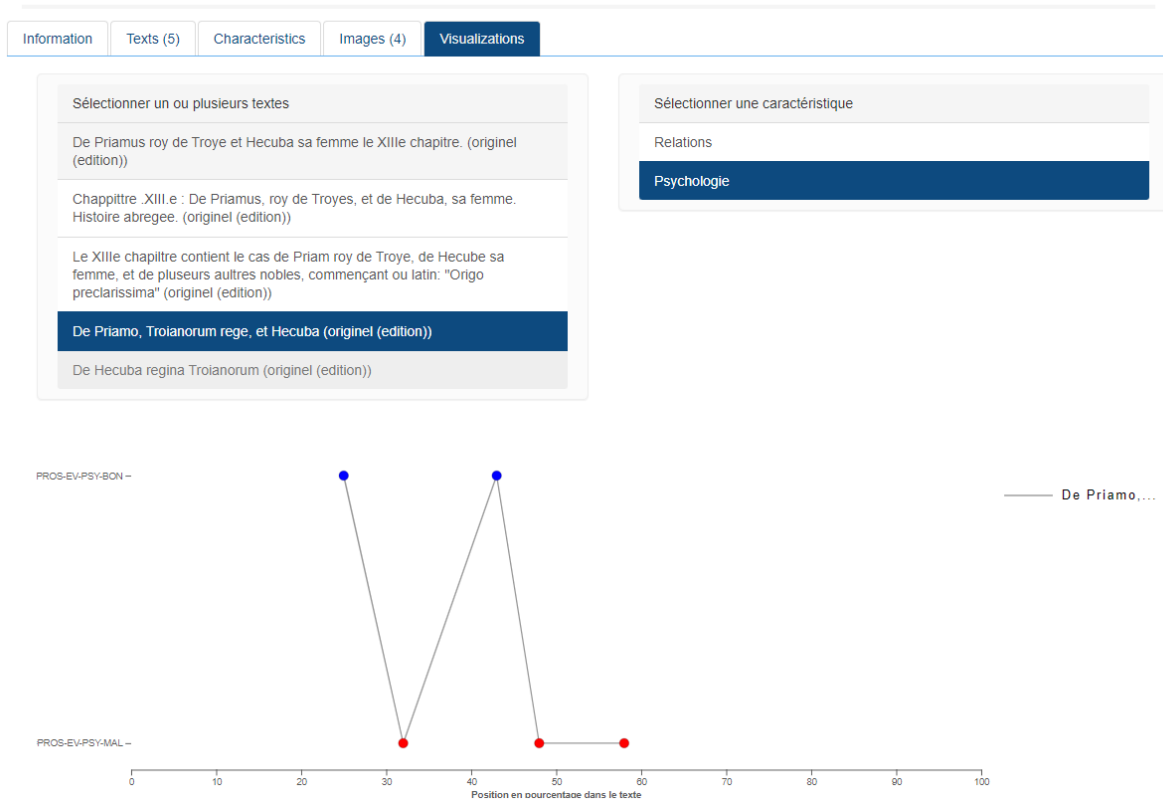


Figure 5.2 – Première version du graphe de l'état psychologique

On peut donc voir par exemple dans la figure ci-dessus que le graphe de l'évolution de l'état psychologique du personnage de Priamus dans le texte “De Priamo, Troianorum rege, et Hecuba” est affiché. Il est considéré comme heureux au début du texte et malheureux vers la fin.

La seconde version du graphe voit arriver la fonctionnalité cœur de ce projet qui est la comparaison de graphe. Il est maintenant possible de sélectionner plusieurs textes et de superposer les courbes afin de pouvoir plus facilement les comparer entre elles. Chaque courbe est d'une couleur différente et le graphe se voit maintenant accompagné d'une légende afin d'associer les courbes aux textes.

Character: Priamus



Figure 5.3 – Seconde version du graphe de l'état psychologique

La 3eme version correspond à l'ajout d'une tooltip, une bulle d'information qui s'affiche au survol des points d'une courbe par la souris. Ces points (correspondants donc aux annotations du texte) sont maintenant cliquables et ouvrent un nouvel onglet redirigeant vers le texte avec l'annotation surlignée.

Character: Priamus

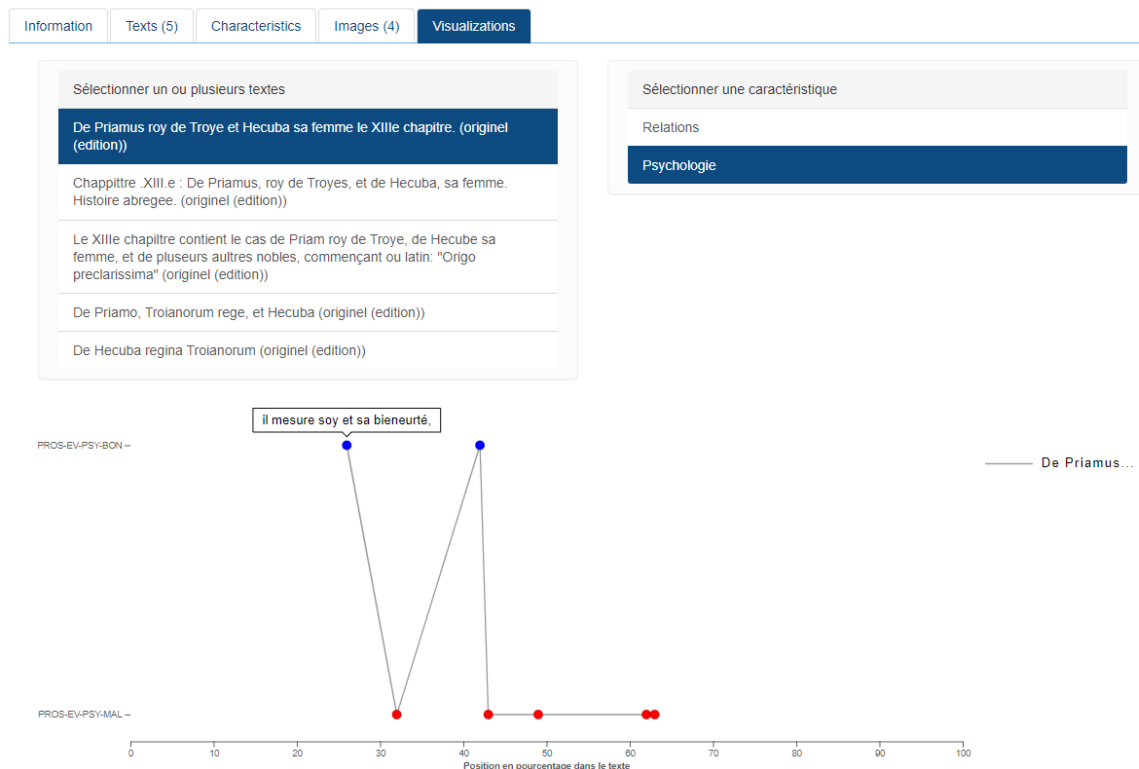


Figure 5.4 – Troisième version du graphe de l'état psychologique

Enfin, la 4eme et dernière version ajoutent aux bulles d'informations précédentes, les jugements de l'auteur. Les jugements de l'auteur sont des annotations supplémentaires pouvant accompagner d'autres annotations. Elles apportent une information supplémentaire sous la forme du point de vue de l'auteur et permettent de mieux analyser la donnée dans son contexte. Une annotation possédant un jugement de l'auteur sera dessinée dans le graphe en tant que carré au lieu de point et la bulle d'information au survol de la souris présentera désormais le jugement en plus de l'annotation.

Character: Priamus

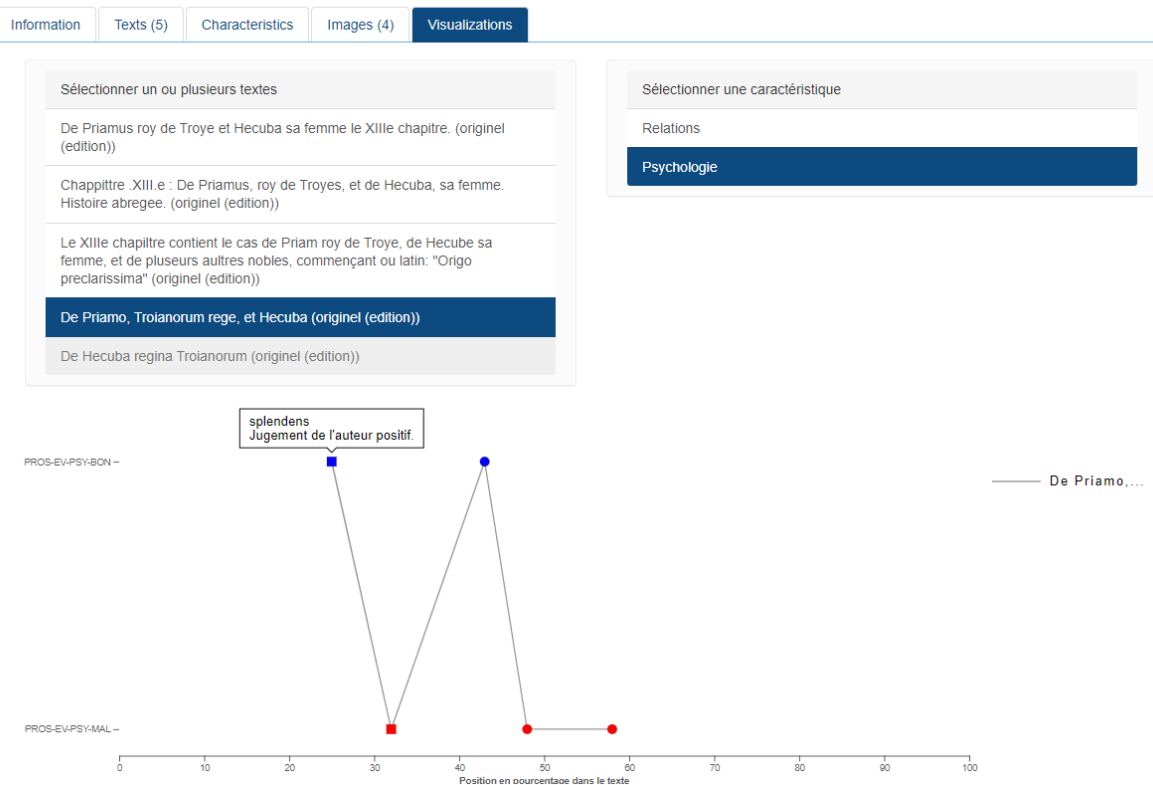


Figure 5.5 – Quatrième version du graphe de l'état psychologique

Le résultat de tout ces développements est la méthode suivante :

```

1 function drawPsychologicalGraph(data) {
2   let width = 1200;
3   let height = 400;
4   let margin = ({top: 30, right: 200, bottom: 30, left: 100});
5   strokeLinecap = "round"; // stroke line cap of the line
6   strokeLinejoin = "round"; // stroke line join of the line
7   strokeWidth = 1.5; // stroke width of line, in pixels
8   strokeOpacity = 1; // stroke opacity of line
9
10  // Construct scales and axes.
11  // The X-Axis is linear and quantitative : from the beginning of the text
12  // (0%) to its end (100%)
13  const xScale = d3.scaleLinear()
14    .domain([0, 100])
15    .range([margin.left, width - margin.right])
16    .interpolate(d3.interpolateRound)
17  // The Y-Axis is qualitative : either #PROS-EV-PSY-BON or #PROS-EV-PSY-MAL
18  const yScale = d3.scalePoint()
19    .domain(["#PROS-EV-PSY-BON", "#PROS-EV-PSY-MAL"])
20    .range([margin.top, height - margin.bottom])
21    .padding(0.1)
22    .round(true)
23
24  const xAxis = d3.axisBottom(xScale);
25  const yAxis = d3.axisLeft(yScale);
26
27  // The main svg node
28  const svg = d3.create("svg")
29    .attr("width", width)
30    .attr("height", height)

```

```

30         .attr("viewBox", [0, 0, width, height])
31         .attr("style", "max-width: 100%; height: auto;");
32 // The X-axis
33 svg.append("g")
34   .attr("transform", `translate(0, ${height - margin.bottom})`)
35   .call(xAxis)
36   .call(g => g.append("text")
37     .attr("x", 460)
38     .attr("y", margin.bottom)
39     .attr("fill", "currentColor")
40     .attr("text-anchor", "start")
41     .attr("font-weight", "bold")
42     .text("Position en pourcentage dans le texte"));
43 // The Y-Axis
44 svg.append("g")
45   .attr("transform", `translate(${margin.left},0)`)
46   .call(yAxis)
47   .call(g => g.select(".domain").remove())
48
49 // We draw each lines and set of points/annotations individually
50 let iter = 0;
51 data.forEach( (value, key, map) => {
52   color = d3.schemeSet1[8 - iter]; // stroke color of line, starting from
53   // the end because I prefer the colors
54   if (iter < 7) {
55     iter = iter + 1;
56   }
57
58   // Compute values.
59   const X = d3.map(value, d => d.get("position"));
60   const Y = d3.map(value, d => d.get("annotation_code"));
61   const I = d3.range(X.length);
62
63   // Construct a line generator.
64   const line = d3.line()
65     .curve(d3.curveLinear)
66     .x(d => xScale(X[d]))
67     .y(d => yScale(Y[d]));
68
69   // The line connecting the dots
70   svg.append("path")
71     .attr("fill", "none")
72     .attr("stroke", color)
73     .attr("stroke-width", strokeWidth)
74     .attr("stroke-linecap", strokeLinecap)
75     .attr("stroke-linejoin", strokeLinejoin)
76     .attr("stroke-opacity", strokeOpacity)
77     .attr("d", line(I))
78     .style("pointer-events", "none");
79
80   let value_author_judgement = value.filter( element => element.get("
81     author_judgement") != null );
82   let value_not_author_judgement = value.filter ( element => element.get("
83     author_judgement") == null );
84
85   // The dots corresponding to the annotations in the text
86   svg.append("g")
87     .attr("stroke", "#000")
88     .attr("stroke-opacity", 0.2)
89     .selectAll("circle")
90     .data(value_not_author_judgement)
91     .join("circle")
92     .attr("cx", d => xScale(d.get("position"))) // We set the
93     // positions thanks to the scales which give us the absolute
94     // values.

```

```

89         .attr("cy", d => yScale(d.get("annotation_code")))
90         .attr("fill", d => (d.get("annotation_code") === "#PROS-EV-PSY-
          BON") ? "blue" : "red")
91         .attr("r", 5)
92         .style("cursor", "pointer")
93     // The squares corresponding to the annotations in the text accompanied
        by an author's judgement
94     svg.append("g")
95     .attr("stroke", "#000")
96     .attr("stroke-opacity", 0.2)
97     .selectAll("rect")
98     .data(value_author_judgement)
99     .join("rect")
100         .attr("offset-anchor", "center")
101         .attr("x", d => (xScale(d.get("position")) - 5)) // We set the
            positions thanks to the scales which give us the absolute values
102
103         .attr("y", d => (yScale(d.get("annotation_code")) - 5))
104         .attr("fill", d => (d.get("annotation_code") === "#PROS-EV-PSY-BON")
            ? "blue" : "red")
105         .attr("width", 10)
106         .attr("height", 10)
107         .style("cursor", "pointer")
108
109     // The legend
110     svg.append("line")
111     .attr("fill", "none")
112     .attr("stroke", color)
113     .attr("stroke-width", strokeWidth)
114     .attr("stroke-linecap", strokeLinecap)
115     .attr("stroke-linejoin", strokeLinejoin)
116     .attr("stroke-opacity", strokeOpacity)
117     .attr("x1", 1025)
118     .attr("x2", 1075)
119     .attr("y1", margin.top + iter * 50)
120     .attr("y2", margin.top + iter * 50)
121
122     svg.append("text")
123     .attr("x", 1085)
124     .attr("y", margin.top + 4 + iter * 50)
125     .attr("textLength", 100)
126     .text(key.slice(0, 10).concat("..."))
127
128     // The tooltip appearing on mouseover
129     // We draw it last because we want it over every other element (lines and
        dots)
130     const tooltip = svg.append("g")
131     .style("pointer-events", "none");
132
133     // The use of ".bind()" is one of the only way to pass parameters to an
        event handler, it replace the result of the "this" keyword with the
        value of the argument
134     svg.selectAll("circle")
135     .on("mouseenter", mouseEntered.bind({"data" : data, "xScale" : xScale, "
        yScale" : yScale, "tooltip" : tooltip}))
136     .on("mouseleave", mouseLeft.bind({"tooltip" : tooltip, "svg" : svg}))
137     .on("click", click.bind({"data" : data}));
138
139     svg.selectAll("rect")
140     .on("mouseenter", mouseEntered.bind({"data" : data, "xScale" : xScale, "
        yScale" : yScale, "tooltip" : tooltip}))
141     .on("mouseleave", mouseLeft.bind({"tooltip" : tooltip, "svg" : svg}))
142     .on("click", click.bind({"data" : data}));

```

```

143 |
144 |     d3.select("#graph").node().append(svg.node());
145 | }

```

Dans les grandes lignes, cette méthode suit un algorithme relativement simple :

- Dans un premier temps, l'élément svg principal est créé et les axes sont construits et ajoutés à ce dernier. L'axe des abscisses est un axe linéaire et quantitatif, représentant le pourcentage de complétion d'un texte (0% étant le début du texte et 100% sa fin). L'axe des ordonnées est un axe qualitatif, contenant deux valeurs, les annotations "heureux" et "malheureux".
- Ensuite, pour chaque texte passé en paramètre :
 - La ligne qui connecte les points est créée.
 - Ainsi que les dits points : pour chaque annotation du texte, si elle ne comporte pas de jugement de l'auteur, alors le point est un cercle, sinon un carré. Le point est de couleur bleue si l'annotation représente un bonheur et rouge si c'est un malheur.
 - Et enfin, on ajoute à la légende la couleur correspondant à la ligne avec le titre du texte qu'elle représente
- Et pour finir, la bulle d'information apparaissant au survol de chaque point est créée. Par défaut, cette bulle d'information affiche le texte de l'annotation et lorsque cette dernière est accompagnée d'un jugement de l'auteur, elle affiche aussi ce dernier.

6

Bilan et conclusion

1 Bilan du semestre 9

Au cours de ce semestre 9, nous avons pu cadrer les besoins ainsi que modéliser et spécifier les solutions au cours de réunions avec de nombreux acteurs au sein du projet, les maquettes ont été réalisées et validées, le cahier des spécifications a été rédigé et validé et nous avons même pu commencer à nous attaquer à certaines problématiques et à préparer le terrain pour le semestre 10 (voir problèmes de l'absence de marqueur temporel sur les données). J'ai donc déjà pu mettre les mains dans le code et me familiariser avec l'environnement.

Pour ce qu'il reste à faire, c'est avant tout la réalisation même. Au niveau de l'API le gros du travail est sur la mise au point de l'ontologie, dont la tâche revient à ma binôme Marie-Gabrielle tandis que je n'aurais qu'à développer l'API, l'environnement qui va accueillir l'ontologie. Tandis que pour les outils de visualisations : la création des écrans, c'est-à-dire concrétiser les interfaces présentées dans les maquettes et évidemment l'implémentation des fonctions créant les graphes, les représentations visuelles des données. Sans oublier la rédaction de la documentation tout au long des développements ainsi que la tenue des tests.

En conclusion de ce 9ème semestre, je peux dire que le bilan est plutôt positif, nous avons bien avancé malgré les problèmes auxquels nous avons fait face, nous avons su résoudre ces derniers. Je suis satisfait de la coopération avec Marie-Gabrielle, j'estime que nos panels de compétences s'équilibrent bien et que nous pouvons chacun apporter notre expertise sur des sujets assez différents.

J'ai aussi découvert le domaine de l'humanité numérique ainsi que le web sémantique ainsi que ses bonnes pratiques qui l'accompagnent.

2 Bilan sur la qualité

Lorsque l'on revient sur tout le travail réalisé, force est de constater que tous les objectifs n'ont pas été remplis et que le graphe de relations entre personnages n'a pas été livré. Malgré cet échec, je reste tout de même satisfait de la qualité de ce qui a été rendu. J'estime que le code produit est assez propre et bien commenté pour être facilement étendu à d'autres développements si mon travail venait à être repris dans le futur.

3 Bilan autocritique sur la gestion

La spécificité de ce projet ayant été le travail en binôme pour toute la durée du 1er semestre, nous avons su en tirer profit avec une bonne répartition des tâches en accord avec nos domaines de compétences. Nous avons aussi assuré une bonne communication avec le client tout au long de ce projet, de la délimitation des besoins et sa spécification à ses débuts jusqu'au suivi des développements jusqu'à la fin. Le principal point noir du déroulement de ce projet se trouve dans le chiffrage des différentes tâches. En effet, je n'ai pas eu une vision assez précise, je n'ai pas assez divisé certaines tâches ce qui a amené à plein de petits développements imprévus qui au final ont consommé beaucoup de temps de travail. Néanmoins je suis assez satisfait de l'efficacité avec laquelle j'ai pu m'approprier un projet existant et y ajouter des fonctionnalités à l'aide d'un outil que j'ai dû apprendre sur le tas.

Annexes

A

Gestion de projet

1 Planification

Le diagramme de Gantt Initial pour la planification de ce projet

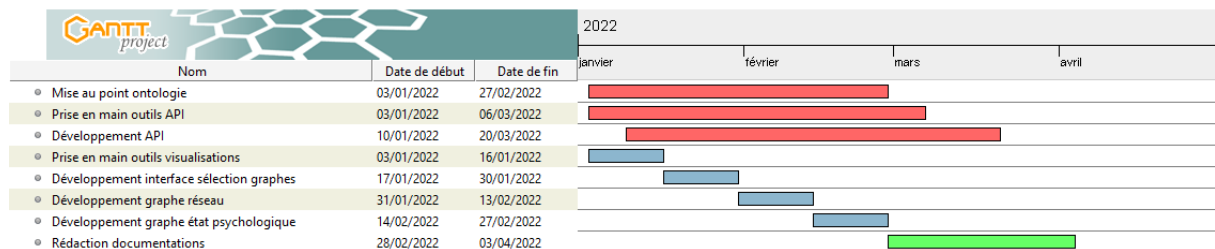


Figure A.1 – Le diagramme de Gantt initial prévu pour le semestre 10

Le diagramme de Gantt final à la fin du projet

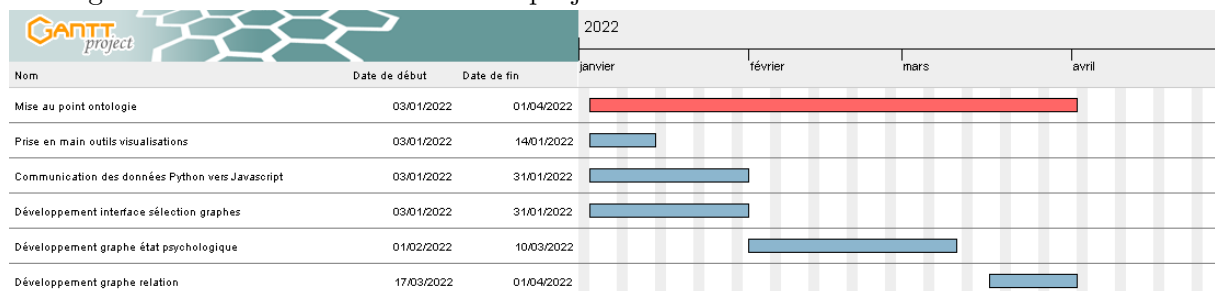


Figure A.2 – Le diagramme de Gantt final à la fin du semestre 10

2 Description des tâches

Tâche 1 : Mise au point ontologie

- Date de début : 03/01/2022
- Date de fin : 27/02/2022
- Durée : 56 jours

- Description : Mise au point de l'ontologie RDF qui servira d'entrepôt de données à l'API. La période de réalisation est aussi longue car il est difficile de chiffrer un temps précis pour cette tâche, nous mettrons donc à jours notre planning de projet au fil des semaines pour y intégrer le développement de l'API lorsque nous aurons plus de visibilité sur la complétion de l'ontologie.
- Ressource assignée : Marie-Gabrielle

Tâche 2 : Prise en main outils API

- Date de début : 03/01/2022
- Date de fin : 06/03/2022
- Durée : 63 jours
- Description : Prise en main des technologies qui serviront à développer l'API afin de garantir de bon fondamentaux et de ne pas perdre de temps par la suite. Nécessite que la réalisation de l'ontologie touche au moins à sa fin.
- Ressource assignée : Olivier

Tâche 3 : Développement de l'API

- Date de début : 10/01/2022
- Date de fin : 20/03/2022
- Durée : 70 jours
- Description : Développement de l'API même. Nécessite la complétion de l'ontologie et la prise en main des outils de développement.
- Ressource assignée : Olivier

Tâche 4 : Prise en main des outils de visualisations

- Date de début : 03/01/2022
- Date de fin : 16/01/2022
- Durée : 14 jours
- Description : Prise en main des technologies qui serviront à développer les outils de visualisations afin de garantir de bon fondamentaux et de ne pas perdre de temps par la suite.
- Ressource assignée : Olivier

Tâche 5 : Développement des interfaces de sélection des graphes

- Date de début : 17/01/2022
- Date de fin : 30/01/2022
- Durée : 14 jours
- Description : Développement des interfaces de sélection des graphes à afficher, conformément aux maquettes établies, en fonction d'un ou de plusieurs textes.
- Ressource assignée : Olivier

Tâche 6 : Développement de l'affichage du graphe du réseau social

- Date de début : 31/01/2022
- Date de fin : 13/02/2022
- Durée : 14 jours
- Description : Développement de la fonction permettant d'afficher la représentation de l'évolution du réseau social d'un personnage. Nécessite la complétion de l'interface de sélection des graphes.
- Ressource assignée : Olivier

Tâche 7 : Développement de l’affichage du graphe de l’état psychologique

- Date de début : 14/02/2022
- Date de fin : 27/02/2022
- Durée : 14 jours
- Description : Développement de la fonction permettant d’afficher la représentation de l’évolution de l’état psychologique d’un personnage.
Nécessite la complétion de l’interface de sélection des graphes.
- Ressource assignée : Olivier

Tâche 8 : Rédaction des documents projet

- Date de début : 28/02/2022
- Date de fin : 03/04/2022
- Durée : 35 jours
- Description : Rédaction des divers documents liés au projet : les documentations fonctionnelle et d’installation, le cahier de développeur, le dossier de test mais aussi le rapport et la préparation à la soutenance.
- Ressource assignée : Olivier & Marie-Gabrielle

B

Cahier de Spécifications



ÉCOLE POLYTECHNIQUE DE L'UNIVERSITE FRANÇOIS RABELAIS DE TOURS
Spécialité Informatique
64 av. Jean Portalis
37200 TOURS, FRANCE
Tél +33 (0)2 47 36 14 31
www.polytech.univ-tours.fr

[illegible]

TABLE DES MATIERES

Cahier de spécifications	3
1. Introduction	3
2. Contexte de la réalisation	3
3. Objectifs	4
4. API	4
4.1. Description générale	4
4.2. Description des interfaces externes du logiciel.....	5
4.3. Spécifications fonctionnelles.....	5
4.4. Spécifications non fonctionnelles	6
5. Outils de visualisation	7
5.1. Description générale	7
5.2. Description des interfaces externes du logiciel.....	8
5.3. Spécifications fonctionnelles.....	13
5.4. Spécifications non fonctionnelles	14
Glossaire.....	16
Bibliographie	17

1. Introduction

Ce document présente les objectifs, le contexte et les spécifications techniques du projet recherche et développement « Outils pour l'exploitation de la prosopographie dynamique : ontologie et visualisations ». Ce projet étant construit en deux phases indépendantes répondant chacune à un besoin particulier, ce cahier de spécification les couvrira de la même manière, en deux parties distinctes.

La maîtrise d'ouvrage est représentée par Elena PIERAZZO, enseignante-chercheuse en humanité numérique au Centre d'Etudes Supérieures de la Renaissance.

La maîtrise d'œuvre est représentée par Olivier MILLOCHAU, étudiant en 5ème année à Polytech Tours, département informatique, encadré par Mohamed SLIMANE enseignant à Polytech Tours, département informatique, et Marie-Gabrielle LAGASSE, étudiante en 5ème année en humanité numérique au Centre d'Etudes Supérieures de la Renaissance, auteures de ce cahier des spécifications.

2. Contexte de la réalisation

Le projet BONHum, BOccace Numérique Humaniste, est une Édition numérique des œuvres de Boccace qui consiste à étudier la prosopographie et la sémantique au sein des textes. Ce travail a pour but de faire ressortir l'essence humaniste qui jalonne les écrits de l'auteur et vise à mettre en lumière le rôle fondamental de Boccace (1313-1375) pour la diffusion de l'Humanisme dans l'espace culturel européen (Bonhum, s.d.).

Les nombreux ouvrages de Boccace sont bien le véhicule privilégié de la nouvelle sensibilité humaniste en Europe. Ainsi, pour exposer avec clarté le caractère humaniste des écrits de Boccace, une collaboration scientifique internationale, des outils d'analyse et de valorisation, ainsi que la création d'une archive incrémentale de textes ont été mis en place. Ce projet étant un travail de recherche, il est destiné à s'élargir en termes de textes, images et partenaires (Bonhum, s.d.).

Le projet BONHum débute en janvier 2019 avec la volonté première de créer un modèle d'édition numérique des textes de Boccace, écrits en latin et en vulgaire, ainsi que ses traductions françaises (XVe-XVIe siècles). Cette opération de numérisation vise à répondre à la question suivante : comment dégager le caractère humaniste de l'œuvre de Boccace ?

Actuellement, parmi les œuvres de Boccace, le travail d'édition numérique a été effectué pour :

- 16 textes du *Decameron*
- 12 textes du *De mulieribus claris*
- 9 textes de *De casibus vivorum illustrium*
- 3 textes des traductions françaises de 1400 et 1409 du *De casibus vivorum illustrium* traduit en français par *Des cas des nobles hommes et femmes*

À cela s'ajoute :

- Un travail de balisage en XML des textes
- Les annotations d'images dans une base de données
- Les données prosopographiques (une partie en base de données et une partie en XML) accessibles via un système de filtrage

3. Objectifs

L'objectif de cette collaboration est de développer des outils pour l'exploitation de la prosopographie dynamique. Pour ce faire, les deux objectifs principaux sont les suivants :

- Créer une ontologie permettant de regrouper la base de données relationnelles (BDR) et les fichiers XML des textes de Boccace présents sur le site de BonHum afin de créer un modèle de données qui soit formel (modèle RDF/OWL).
Ce modèle de donnée sera ensuite destiné à être mis en ligne pour être exploitable par le plus grand monde au travers d'une API Linked Open Data.
- Réfléchir à des outils de visualisation des données prosopographiques pour représenter l'évolution socio-professionnelle et psychologique du personnage en fonction de la temporalité du récit.

Concernant le second point, toute la difficulté se trouve dans l'idée de lier ensemble les mutations, les évolutions, les changements que peut avoir un même personnage au cours du temps. En effet, un même personnage accède dans son développement à certaines caractéristiques qui lui sont propres et qui sont souvent dues aux rencontres qu'il fait, aux occasions qui se présentent ou pas à lui et qui impactent sa vie au cours du récit. La complexité de cette représentation se situe par les occurrences de ces rencontres et de ces états. En effet, le personnage peut être amené à rencontrer à plusieurs reprises un autre personnage, mais sur une échelle de temps de différentes et avec un statut social et une fonction différente. De même au cours de sa vie, il peut changer constamment d'état passant ainsi d'heureux à malheureux et de malheureux à heureux. Il faut trouver une manière de représenter ces évolutions dans le temps de manière fluctuante et de ne pas tomber dans l'immuable et l'invariant. On cherche à donner de la profondeur et de l'épaisseur et à montrer des niveaux d'évolution et des couches chronologiques.

Ce projet se constitue donc en deux phases, premièrement, la réalisation d'un tout nouveau système d'informations sous la forme d'une API, fonctionnant indépendamment de l'application Bonhum et deuxièmement, la mise au point de divers outils de visualisations de données fournis par cette précédente API.

4. API

4.1. Description générale

4.1.1. Environnement du projet

L'API ayant pour objectif de fonctionner indépendamment de l'application Bonhum, elle ne s'intégrera pas dans son environnement et en disposera d'un qui lui est propre.

On retrouve dans cet environnement une ontologie mise au point par Marie-Gabrielle. C'est cette ontologie qui servira d'entrepôt de données aux données des objets « Personnages », déjà présent dans l'application Bonhum et qui fournira ces dernières au format RDF à l'API.

L'API sera développée avec **Python** comme langage de programmation associé au framework **Django**. Seront aussi utilisés le framework **Django Rest Framework**, outil très puissant pour créer des API web et la librairie **sparql-client** pour générer des requêtes SparQL et communiquer avec l'ontologie. Tous ces outils seront dans leur dernière version à l'heure où est rédigé ce document.

Les développements se feront dans l'éditeur de texte **Visual Studio Code**.

Une fois l'API terminée, elle sera vouée à être hébergée sur un serveur de l'université.

4.1.2. Caractéristiques des utilisateurs

De par sa nature, l'API est avant tout destinée à être utilisée par des développeurs.

Par contre, les données qu'elle fournira sont elles à l'attention de chercheurs en humanité numérique, d'où l'importance du respect du format de données RDF, il est nécessaire que les données soient inter-opérables. Enfin, il n'y a aucun droit d'accès à l'API dû à son standard « Linked Open Data »

4.1.3. Fonctionnalités du système

En tant qu'API Web Linked Open Data, son unique fonction sera de fournir les données des personnages des œuvres de Boccace par le biais de requêtes HTTP.

Deux requêtes seront possibles :

- Une qui fournira les données d'un personnage en particulier.
- Une autre qui fournira les données de tous les personnages, avec possibilité d'y ajouter divers filtres.

4.1.4. Structure générale du système

L'architecture de notre système correspondra à celle d'une API Django Rest Framework classique.

On y retrouvera 2 blocs majeurs :

- Le « ViewSet » : Ils sont responsables du traitement de demandes spécifiques, on peut les retrouver sous le nom de « Controller » dans d'autres framework.
Dans notre API, on retrouvera un unique ViewSet (dédié à l'objet « Personnage ») qui comportera deux méthodes, chacune dédiée à une des deux requêtes présentées plus haut.
- Le « Router » : Détermine à quel « ViewSet » transmettre une requête entrante.

A cette architecture s'ajoute l'ontologie RDF-OWL mise au point par Marie-Gabrielle, source de données au format RDF des entités « Personnages » dont l'API servira de point d'ouverture sur le web.

4.2. Description des interfaces externes du logiciel

4.2.1. Interfaces matériel/logiciel

De par sa nature l'API, l'application aura besoin d'un serveur ouvert sur le web pour fonctionner.

4.2.2. Interfaces homme/machine

L'interaction entre les utilisateurs et l'API se fera principalement par requêtage et réponse HTTP, il n'y a donc pas d'interface visuelle.

4.2.3. Interfaces logiciel/logiciel

L'ontologie fera office d'entrepôt de données où seront stockées toutes les données relatives aux objets « Personnages » déjà présents dans l'application Bonhum (originaires là, de sa base de données et des textes annotés).

L'API sera ensuite capable de requêter l'ontologie grâce au langage SparQL et de moissonner les données au format RDF. Elle joue donc le rôle ici d'interface entre le web et l'ontologie.

4.3. Spécifications fonctionnelles

4.3.1. Définition de la fonction 1 : getOneCharacter

Identification de la fonction

Cette fonction a pour but de fournir les données d'un personnage en particulier.

Description de la fonction

Cette fonction est appelée suite au requêtage d'un utilisateur à l'API d'un personnage en particulier, identifié par son nom.

Une requête SparQL sera construite pour récupérer les données du personnage voulu puis transmise à l'ontologie. La réponse de cette dernière renvoyée à l'utilisateur.

En cas de nom invalide, une réponse contenant un code d'erreur sera renvoyée.

4.3.2. Définition de la fonction 2 : getAllCharacter

Identification de la fonction

Cette fonction a pour but de fournir les données de tous les personnages.

Description de la fonction

Cette fonction est appelée suite au requêtage d'un utilisateur à l'API de tous les personnages.

Divers filtres pourront être appliqués, correspondant à ceux déjà présent dans l'application Bonhumn soit :

- Par « Type ».
- Par « Sexe ».
- Par « Age ».
- Par « Religion ».
- Par « Titre ».
- Par « Origine Géographique ».
- Par « Traits ».
- Par « Profession ».
- Par « Statut marital ».
- Par « Condition psychologique ».
- Par « Changement de socio-économique ».
- Par « Niveau socio-économique ».
- Par « Niveau de culture ».
- Par « Relation familiale ».
- Par « Type de relations sociales ».
- Par « Hiérarchie de relations sociales ».
- Par « Lieux ».

Une requête SparQL sera construite pour récupérer les données de tous les personnages en prenant en compte les filtres voulus puis transmise à l'ontologie. La réponse de cette dernière renvoyée à l'utilisateur.

En cas de filtre invalide, une réponse contenant un code d'erreur sera renvoyée.

4.4. Spécifications non fonctionnelles

4.4.1. Contraintes de développement et conception

Afin de satisfaire les principes du « Linked Open Data », l'API doit en partie fournir des données au format RDF. C'est pour cela que nous devons utiliser le langage de requête SparQL.

L'utilisation de ce langage passera par la librairie python **sparql-client** qui permet de requêter des points de terminaisons de communication (ici, l'ontologie).

Tout au long du développement, une documentation technique et fonctionnelle exhaustive sera rédigée. On y retrouvera un mode d'emploi sur le déploiement de l'API ainsi que sur son utilisation. Le code sera lui aussi suffisamment commenté.

Des tests unitaires de l'application seront assurés par le module python **unittest** et grâce aux infrastructures de tests de Django et de Django Rest Framework.

4.4.2. Contraintes de fonctionnement et d'exploitation

Performances

Aucune demande particulière sur les performances du système n'a été émise.

Capacités

Aucune demande particulière sur la capacité du système n'a été émise.

Contrôlabilité

Aucune mesure de contrôlabilité particulière du système n'a été émise.

Sécurité

Aucun niveau de confidentialité du système prévu, l'API respectant les principes Linked Open Data, elle est ouverte au web.

5. Outils de visualisation

5.1. Description générale

5.1.1. Environnement du projet

Ces outils de visualisation s'intègrent au sein de l'application Bonhum.

L'environnement du projet va donc dépendre de celui de l'application, c'est-à-dire que les développements se font en **python** version 2.7 sur le framework **Django** en version 1.8 et en **Javascript**. Nous utiliserons en plus la librairie **D3.js** pour créer les visualisations mêmes. L'éditeur de texte **Visual Studio Code** sera utilisé pour développer.

Les outils visent à représenter l'évolution d'objets « Personnage » et de leurs composantes, dans un premier temps, leur statut socio-économique et leur état psychologique. Les données de personnages sont contenues dans une base de données PostgreSQL et dans des listes XML. La manipulation de ces dernières se fera au travers de requêtes déjà implémentées dans l'application.

5.1.2. Caractéristiques des utilisateurs

Ces outils sont destinés à une utilisation régulière par des chercheurs en humanité numérique qui possèdent déjà une certaine connaissance de l'application existante.

Aucun droit d'accès particulier ne contraindre l'utilisation des outils de visualisation, tous les utilisateurs pouvant accéder à l'application peuvent accéder aux outils.

5.1.3. Fonctionnalités du système

L'objectif est d'afficher une représentation visuelle de l'évolution d'une composante d'un objet « Personnage » au cours du temps pour un ou plusieurs textes donnés (à des fins de comparaisons de textes).

Lorsque l'utilisateur est sur la page d'un personnage, il a la possibilité d'accéder à un onglet « Visualisation ». Une fois dans cet onglet, il est possible de sélectionner une composante ainsi qu'un ou plusieurs textes, parmi tous ceux où apparaît le personnage. Une fois la sélection réalisée, un graphe s'affiche. Il est possible sur certaines représentations, d'avancer ou de reculer dans le temps grâce à un curseur défilant.

5.1.4. Structure générale du système

La portée du système va se limiter à deux fichiers.

Le premier, « record_characters.html », est la page HTML même de consultation d'un personnage.

Elle sera amenée à être modifiée pour accueillir un nouvel onglet « Visualisations » qui contiendra notre système de visualisations de personnages.

On retrouvera dans cet onglet la possibilité de sélectionner un ou plusieurs textes parmi tous ceux liés au personnage ainsi qu'une composante à visualiser (dans un premier temps, les relations sociales et l'état psychologique »).

Sera ensuite affiché un graphe représentant l'évolution de la composante choisie au sein des textes choisis.

Le second fichier sur lequel nous serons amenés à travailler, « visualizations.js », contiendra les méthodes Javascript permettant d'afficher les graphes grâce à la librairie D3.js.

5.2. Description des interfaces externes du logiciel

5.2.1. Interfaces homme/machine

Une série de maquette a été réalisée pour illustrer les interfaces et l'expérience utilisateur prévues.

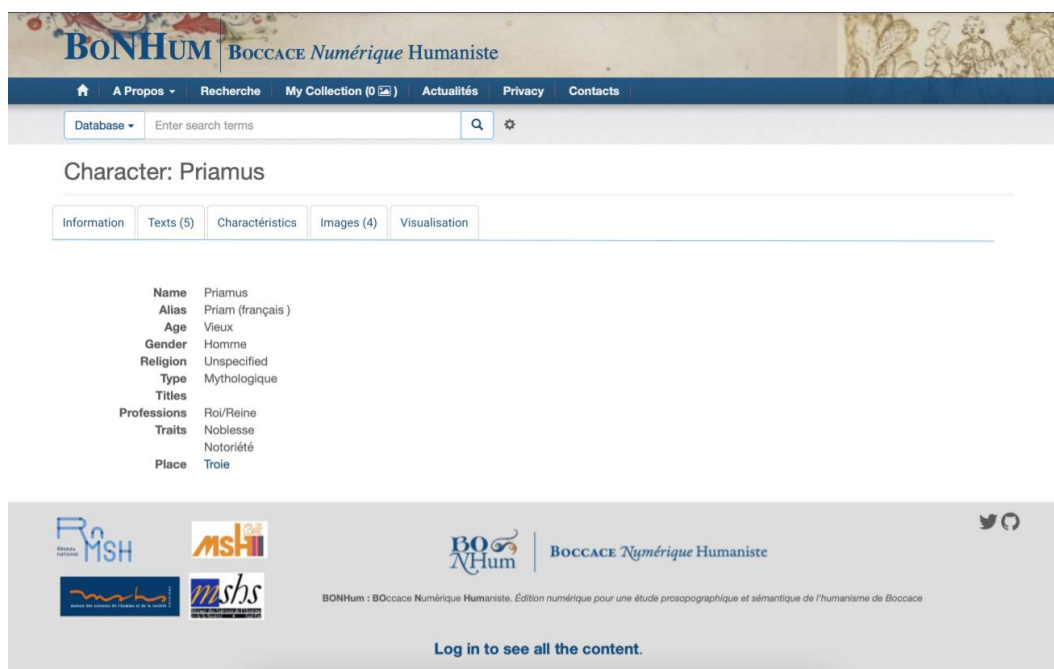


Fig. 1 : Ecran d'accueil de la consultation d'un personnage

Nous pouvons voir dans la figure 1, l'écran d'accueil de la consultation d'un personnage tel qu'il est actuellement avec l'ajout d'un onglet "Visualisation" permettant d'accéder aux outils.

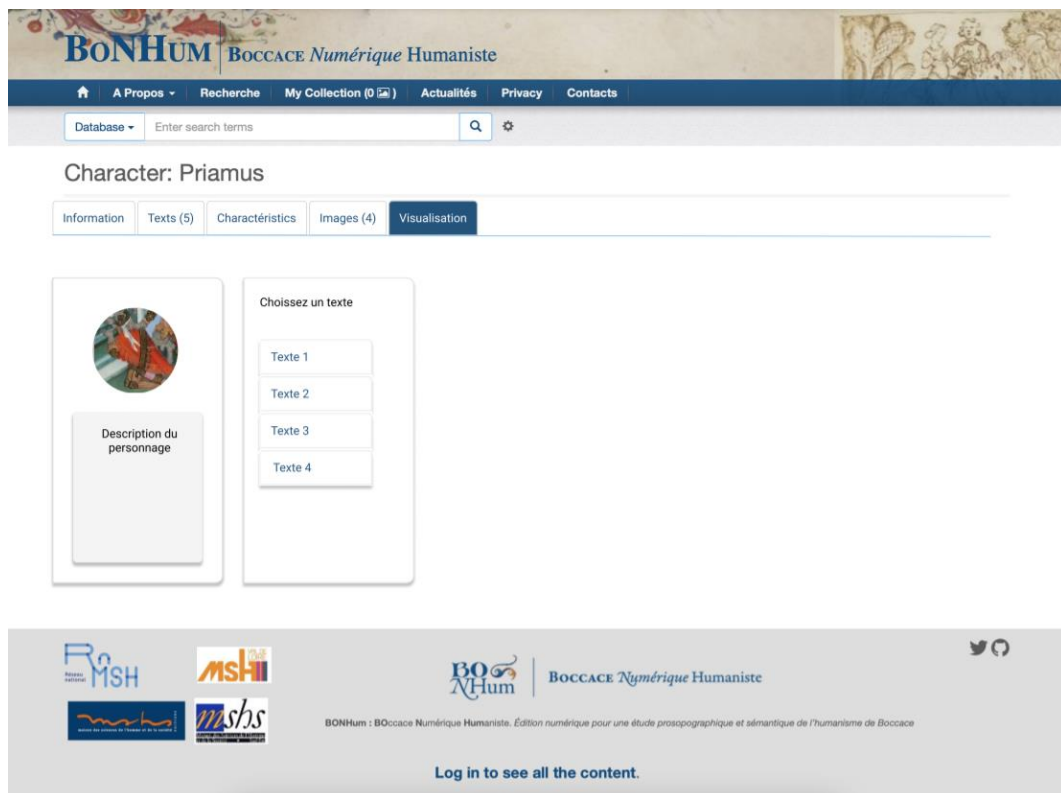


Fig. 2 : Vue d'accueil de l'onglet "Visualisation"

La figure présente l'écran tel qu'il est lorsque l'utilisateur entre dans l'onglet "Visualisation". Il n'y a initialement que la sélection de texte de disponible.

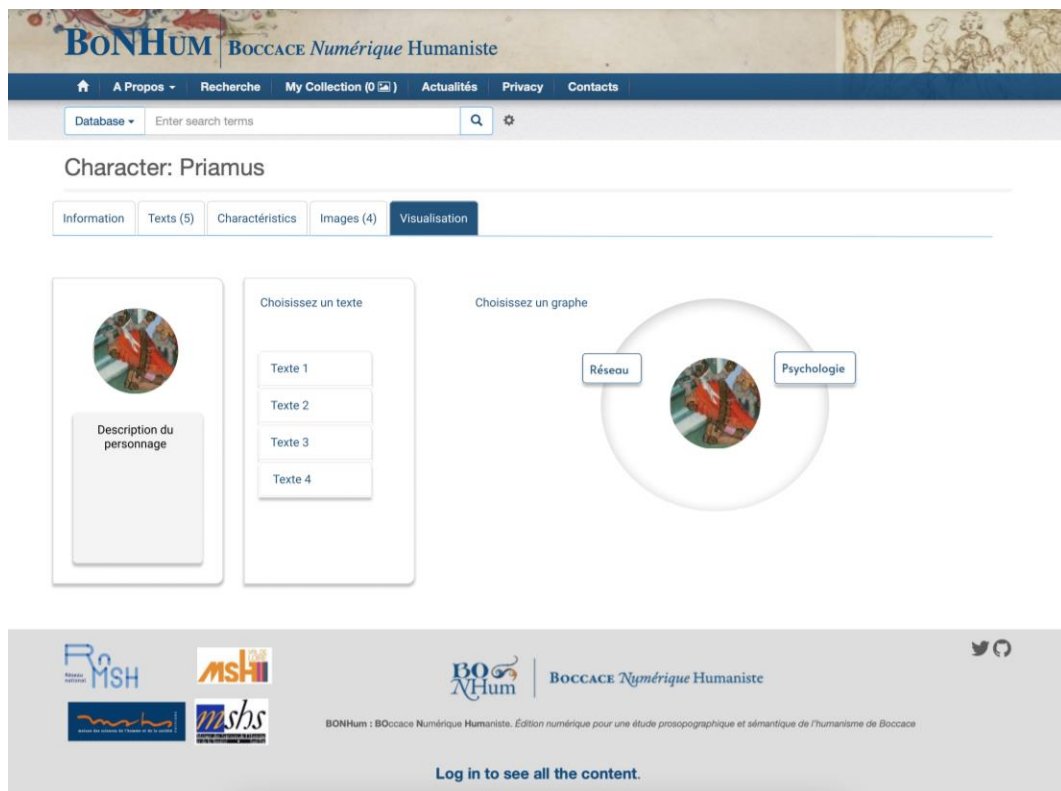


Fig. 3 : Sélection de la composante à représenter

Une fois le ou les textes sélectionnés, la sélection de la composante à représenter s'affiche.
L'idée derrière cette maquette est de proposer une interface intuitive qui accompagne l'utilisateur.



Fig. 4 : Un graphe des relations sociales d'un personnage



Fig. 6 : Défilement du temps au sein de la représentation

Une fois la sélection terminée, le graphe s'affiche.

Ici est représenté le graphe des relations sociales d'un personnage au sein d'un seul texte.

On peut voir dans la figure 5 que ce graphe est interactif : Lorsque l'utilisateur passe sa souris sur une arête, la nature de la relation s'affiche et lorsqu'il la passe au-dessus d'un personnage, une bulle contenant ses informations s'affiche (non représenté ici) et lorsqu'il clique sur un personnage, le graphe de ce dernier s'affiche à la place du graphe courant.

La figure 6 illustre le défilement du temps à l'aide d'un curseur.

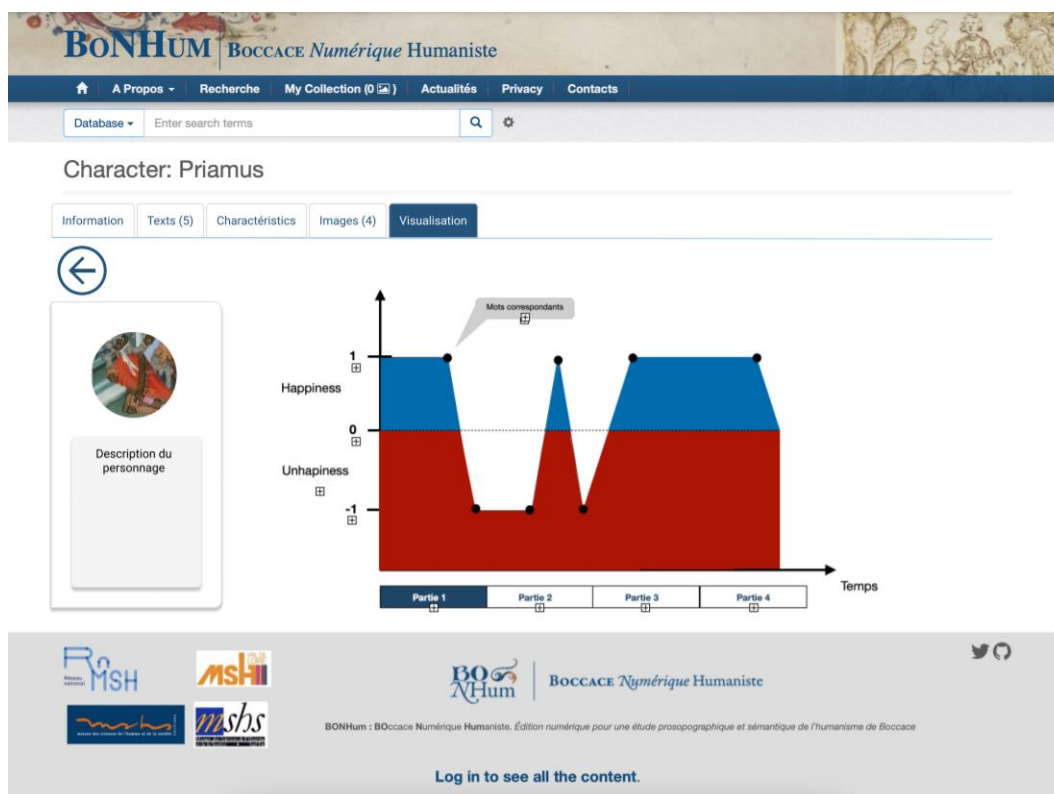


Fig. 8 : Graphe de l'évolution de l'état psychologique

Nous pouvons observer dans cette dernière figure l'autre graphe disponible représentant l'évolution de l'état psychologique d'un personnage au cours d'un texte.

5.2.2. Interfaces logiciel/logiciel

Lorsque que l'utilisateur désire accéder à la page d'un personnage, plusieurs étapes se succèdent menant au chargement des données du personnage voulu :

- Le système reçoit l'information que l'utilisateur tente d'accéder à la page d'un personnage, il transmet la requête à un objet « View ».
- Cet objet va dans un premier temps chargé l'objet personnage depuis la base de données puis va récupérer ses caractéristiques contenues dans des listes XML en retrouvant toutes les annotations (i.e. : balises) liées à ce dernier.
Ces données vont ensuite être transmises à la page HTML de consultation d'un personnage.
- C'est cette page qui va s'occuper d'afficher les données transmises par la « View ».

C'est donc grâce à cette récupération des données du personnage en base de données et dans des listes XML que nous pourrions afficher nos visualisations.

5.3. Spécifications fonctionnelles

5.3.1. Définition de la fonction 1 : drawRelationshipsNetwork

Identification de la fonction

Cette fonction a pour objectif de dessiner un réseau présentant les relations sociales d'un personnage au cours d'un texte.

Chaque personnage sera représenté par un nœud étiqueté du nom du personnage.

Le personnage à visualiser apparaîtra au centre du réseau, il sera entouré et connecté à tous les personnages avec qui il partage un lien social à un moment donnée. Les arêtes seront étiquetées de la nature du lien social. Il sera possible de cliquer sur les nœuds pour afficher le graphe des relations sociales de ce personnage au sein du même texte.

La dimension temporelle peut être manipulée grâce à un curseur défilant sous le graphe.

Description de la fonction

A partir des données du personnage contenues dans le contexte de la page, créer le graphe désiré et l'affiche dans la page.

Il est nécessaire que les données soit marquées temporellement.

5.3.2. Définition de la fonction 1 : drawPsychologicalStatus

Identification de la fonction

Cette fonction a pour objectif de dessiner un graphique présentant l'évolution du statut psychologique d'un personnage au cours d'un texte.

La représentation prendra la forme d'un graphique de différences par rapport à une valeur neutre.

Toute valeur supérieure à cette valeur neutre à un moment donné dénotera du statut heureux pour le personnage et à l'inverse, une valeur inférieure dénotera du statut malheureux.

Description de la fonction

A partir des données du personnage contenues dans le contexte de la page, créer le graphe désiré et l'affiche dans la page.

Il est nécessaire que les données soit marquées temporellement.

5.4. Spécifications non fonctionnelles

5.4.1. Contraintes de développement et conception

Les développements se feront dans l'environnement existant de l'application Bonhum, c'est-à-dire en **python** version 2.7 avec le framework **Django** en version 1.8.

Nous utiliserons en plus la librairie **D3.js**, « une bibliothèque graphique JavaScript qui permet l'affichage de données numériques sous une forme graphique et dynamique. Il s'agit d'un outil important pour la conformation aux normes W3C qui utilise les technologies courantes SVG, JavaScript et CSS pour la visualisation de données » (Wikipedia/D3.js, s.d.). Nous avons choisi cette librairie car elle est l'une des plus puissante et flexible actuellement disponible en termes de manipulation et de représentation des données, condition fondamentale dans ce projet orienté recherche où peu de représentation de ce genre de données existent.

Tout au long du développement, une documentation technique et fonctionnelle exhaustive sera rédigée. Le code sera lui aussi suffisamment commenté.

Des tests de bout en bout seront réalisés pour s'assurer du bon fonctionnement des outils.

5.4.2. Contraintes de fonctionnement et d'exploitation

Performances

Aucune demande particulière sur les performances des outils n'a été émise.

Capacités

Aucune demande particulière sur la capacité des outils n'a été émise.

Contrôlabilité

Aucune mesure de contrôlabilité particulière des outils n'a été émise.

Sécurité

Aucune mesure de sécurité particulière sur les outils n'a été émise.

GLOSSAIRE

API : “Application Programming Interface”, une interface permettant à un système d’offrir ses services à d’autres systèmes. Dans le cadre du Web une API présente le plus souvent des données moissonnables.

Framework : Désigne un ensemble cohérent de composants logiciels structurels, qui sert à créer les fondations ainsi que les grandes lignes de tout ou partie d'un logiciel (Wikipedia/Framework, s.d.).

Linked Open Data : Le Linked Open Data ou « données ouvertes liées » consiste à rendre libre de toute licence des bases de données préalablement liées entre elles selon le modèle du Linked Data de Tim Berners-Lee.

Le Linked Open Data repose sur 4 principes de base essentiels :

- Les données doivent être libres de toute licence pour que des liens soient facilement faits entre elles.
- Chaque lien utilise le modèle Resource Description Framework (RDF).
- Afin d'être localisée, chaque ressource possède un nom de référence, unique et permanent en ligne, Uniform Resource Identifier (URI).
- Les données doivent être mise en ligne suivant le protocole standard HTTP.
(Wikipedia/Linked_Open_Data, s.d.)

Ontologie : Ensemble structuré des termes et concepts représentant le sens d’un champ d'informations, que ce soit par les métadonnées d'un espace de noms, ou les éléments d'un domaine de connaissances (Wikipedia/Ontologie, s.d.).

RDF : Resource Description Framework (RDF) est un modèle de graphe destiné à décrire formellement les ressources Web et leurs métadonnées, afin de permettre le traitement automatique de telles descriptions (Wikipedia//RDF, s.d.).

BIBLIOGRAPHIE

- (s.d.). Récupéré sur Bonhum: <http://bonhum.huma-num.fr>
- (s.d.). Récupéré sur django-rest-framework: <https://www.django-rest-framework.org>
- (s.d.). Récupéré sur Wikipedia/D3.js: <https://fr.wikipedia.org/wiki/D3.js>
- (s.d.). Récupéré sur Wikipedia/Framework: <https://fr.wikipedia.org/wiki/Framework>
- (s.d.). Récupéré sur Wikipedia/Linked_Open_Data: https://fr.wikipedia.org/wiki/Linked_open_data
- (s.d.). Récupéré sur Wikipedia/Ontologie: [https://fr.wikipedia.org/wiki/Ontologie_\(informatique\)](https://fr.wikipedia.org/wiki/Ontologie_(informatique))
- (s.d.). Récupéré sur Wikipedia//RDF: https://fr.wikipedia.org/wiki/Resource_Description_Framework



Bibliographie

- [1] *Boccace (1313-1375) - Bibliographie*. URL : <https://www.bnf.fr/fr/boccace-1313-1375-bibliographie>.
- [2] *BONHum : BOccace Numérique Humaniste*. URL : <http://bonhum.huma-num.fr>.

Outils pour l'exploitation de la prosopographie dynamique: ontologie et visualisations

Olivier MILLOCHAU

Encadrement : Mohamed SLIMANE



En collaboration avec Polytech

Objectifs

La prosopographie constitue la pratique d'inventorier et de classer selon des caractéristiques communes les différentes personnalités qui composent un récit. L'objectif de ce PRD est donc de mettre au point et d'intégrer des outils de visualisations de données prosopographiques au projet BONHum, BOccace Numérique Humaniste, une édition numérique des œuvres de Boccace afin de pouvoir étudier ces dernières dans leur contexte (par exemple, l'impact de différentes traductions sur un même texte).



BOCCACE Numérique Humaniste

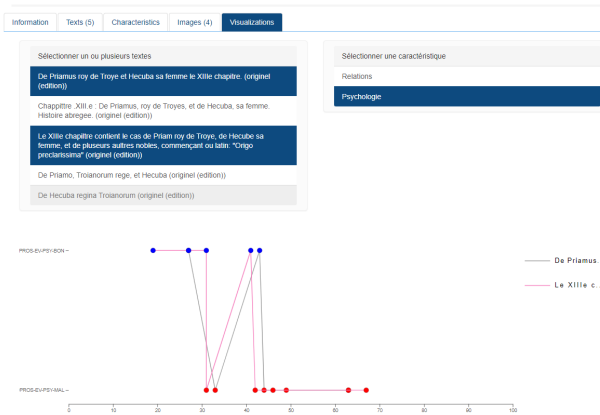
Mise en œuvre

Le projet BONHum héberge actuellement une quarantaine de textes de Boccace, auxquels s'ajoute le travail de prosopographie réalisé via un système d'annotation basé sur du balisage XML. Les outils de visualisations s'appuient sur les données issues de ce travail et sont mis au point grâce à la librairie D3.js.

Résultats

Malheureusement, seulement le graphique de l'évolution de l'état psychologique des personnages aura pu être entièrement complété avant la fin. Néanmoins, toutes les bases ont été posées pour permettre l'expansion du travail réalisé et l'intégration de nouvelles visualisations.

Character: Priamus



Comparaison de l'évolution de l'état psychologique d'un personnage entre deux textes



Outils pour l'exploitation de la prosopographie dynamique: ontologie et visualisations

Olivier MILLOCHAU

Encadrement : Mohamed SLIMANE

Objectifs

La prosopographie constitue la pratique d'inventorier et de classer selon des caractéristiques communes les différentes personnalités qui composent un récit. L'objectif de ce PRD est donc de mettre au point et d'intégrer des outils de visualisations de données prosopographiques au projet BONHum, BOccace Numérique Humaniste, une édition numérique des œuvres de Boccace afin de pouvoir étudier ces dernières dans leur contexte (par exemple, l'impact de différentes traductions sur un même texte).

Mise en œuvre

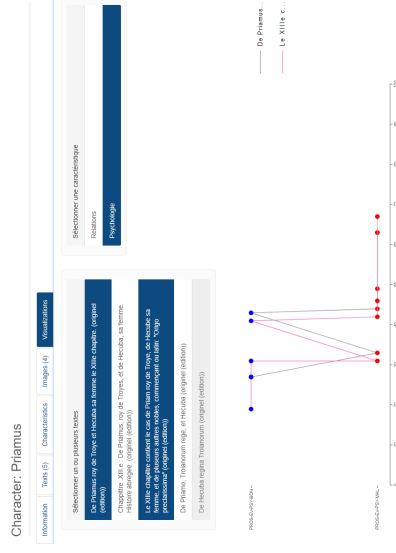
Le projet BONHum héberge actuellement une quarantaine de textes de Boccace, auxquels s'ajoute le travail de prosopographie réalisé via un système d'annotation basé sur du balisage XML. Les outils de visualisations s'appuient sur les données issues de ce travail et sont mis au point grâce à la librairie D3.js.

Résultats

Malheureusement, seulement le graphique de l'évolution de l'état psychologique des personnages aura pu être entièrement complété avant la fin. Néanmoins, toutes les bases ont été posées pour permettre l'expansion du travail réalisé et l'intégration de nouvelles visualisations.



En collaboration avec Polytech



Comparaison de l'évolution de l'état psychologique d'un personnage entre deux textes



Outils pour l'exploitation de la prosopographie dynamique: ontologie et visualisations

Résumé

Ce PRD vise à mettre au point et à intégrer des visualisations de données prosopographiques au sein d'un autre projet existant : le projet BONHum, BOccace Numérique Humaniste, une édition numérique des œuvres de Boccace. Ce rapport couvre toutes les étapes de l'intégration des outils : la prise en main de l'architecture existante et la création de méthodes interface entre cette dernière et les différents ajouts et le développement des graphes même en SVG avec la librairie D3.js.

Mots-clés

Prosopographie, D3.js, Visualisations, Django, Web, SVG

Abstract

This PRD aims to develop and integrate visualizations of prosopographic data within another existing project: the BONHum project, BOccace Numérique Humaniste, a digital edition of Boccace's works. This report covers all the steps of the integration of the tools: learning the existing architecture and creating interfacing methods between it and the new additions and the development of the graphs in SVG with the library D3.js.

Keywords

Prosopography, D3.js, Visualizations, Django, Web, SVG

Entreprise

Polytech



Tuteur entreprise

Elena PIERAZZO

Étudiant

Olivier MILLOCHAU (DI5)

Tuteur académique

Mohamed SLIMANE