Projet Tutoré

M1 Technologies de l'Internet

Conception et développement d'une application d'annotation thématique dans l'environnement Gate

---

Auteurs :
Roland Bary
Charles Follet

Tuteurs :
Marie-Noëlle Bessagnet
Annig Lacayrelle
Albert Royer
Christian Sallaberry

Remerciements

Nous tenons à remercier nos tuteurs pour leur pédagogie et leur encadrement. Monsieur Royer pour sa précision et sa connaissance pointue du domaine. Madame Lacayrelle pour son soutien et sa clarté. Monsieur Sallaberry pour nous avoir remis sur de bonnes pistes quand nous nous égarions. Et enfin, madame Bessagnet pour avoir assuré la coordination et le suivi de ce projet.

Table des matières

Première partie

Introduction

L'évolution des volumes d'informations sur internet provoque une évolution du web vers une approche dans laquelle chaque donnée acquiert un sens afin de rendre possible une interprétation du contenu par des machines. Cette évolution constitue le web sémantique.
Sa principale motivation est la recherche d'information sémantique.

Dans ce cadre, nous sommes intervenus pour répondre à l'appel d'offre de la maîtrise d'ouvrage.
L'objectif correspond à la première étape dans un processus de recherche d'information sémantique et d'indexation : l'annotation sémantique d'un document texte spécifique.

En découle les problématiques suivantes :

- Existe-t-il des outils qui se prêtent aisément à l'annotation sémantique ?
- Quelle approche de conception nous permet de réaliser cette étape d'annotation sur un document texte non-structuré ?

La résolution de ces différentes problématiques nous a amenés à organiser notre réflexion :
Premièrement, nous définirons clairement les demandes et leurs contextes à travers le cahier des charges. Deuxièmement, nous présenterons les connaissances qu'il nous a fallu maîtriser pour ce sujet. Troisièmement, nous détaillerons notre principe de résolution du projet. Nous finirons par le bilan et le retour d'expérience de ce projet.

Deuxième partie

Cahier des charges

I Contexte

A partir des travaux de Georges DEPEYROT sur les monnaies carolingiennes, nous avons travaillé sur l'annotation du Numéraire Carolingien [1].
Sur celui-ci, la maîtrise d'ouvrage a besoin d'effectuer des recherches :

Temporelles : Quelles étaient les pièces en circulation de l'an 859 à l'an 865 ?

Spatiales : Dans quels ateliers, les pièces de type Obole de Charlemagne ont été produites ?

Thématiques : Combien d'exemplaires de la monnaie d'or de Charles le Chauve ont été étudiés ?

Répondre à cette demande implique de définir puis d'explorer les dimensions temporelles, spatiales et thématiques de l'ouvrage.
Pour cela, il est nécessaire de connaître le domaine et l'ouvrage afin de savoir quelle information correspond à quelle dimension.
Une fois cet apprentissage fait, nous pouvons construire des règles dans une chaîne de traitement permettant d'annoter chaque information en fonction de sa dimension.

Les monnaies carolingiennes sont le domaine central pour la réalisation du projet. Les ressources nécessaires à l'annotation (ici sous forme de gazetiers) ont été construites à partir des données de l'ouvrage.

Forte de son expérience dans le domaine, la maîtrise d'ouvrage nous a demandé d'utiliser la boîte à outils logicielle GATE qui sera utile pour le traitement du langage naturel.

En résumé, les caractéristiques du projet sont :
- l'apprentissage et la compréhension du domaine considéré,
- l'étude des principes d'annotation de documents,
- le développement d'une chaîne d'annotation dans GATE,
- la mise en place d'une visualisation des résultats.

1. `http://www.cgb.fr/le-numeraire-carolingien-moneta-77-3e-edition-depeyrot-georges,Ln71,a.html`

II Description

La chaîne de traitement prend un document texte en entrée, produit un document XML en sortie et le met en forme pour une meilleure lisibilité.
Exemple :

Illustrons par un scénario les objectifs de la chaîne de traitement. Elle prend par exemple en entrée le texte suivant

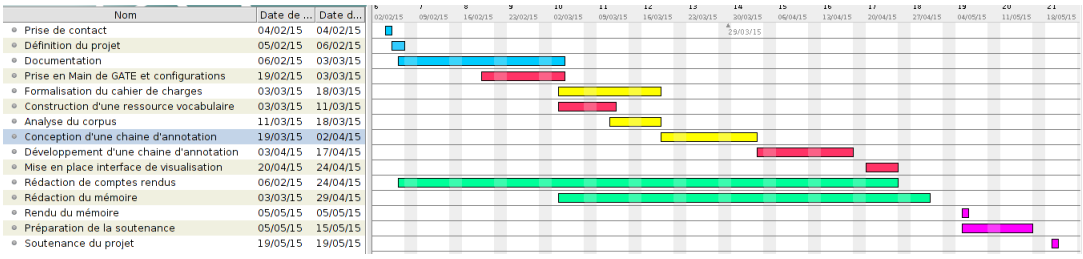
Type de 840-864: Lothaire I (817-855), Pépin II, roi d'Aquitaine (839-865), Charles le Chauve (840-877), Lothaire II roi de Lorraine (855-869), Charles l'Enfant roi d'Aquitaine (vers 860) Denier de Charles le Chauve (43 exemplaires étudiés) + CAROLVS REX FR croix, 4 globes AVTISIODERO CIVI temple

et l'annote

Type de 840-864: Lothaire I (817-855), Pépin II, roi d'Aquitaine (839-865), Charles le Chauve (840-877), Lothaire II roi de Lorraine (855-869), Charles l'Enfant roi d'Aquitaine (vers 860) Denier de Charles le Chauve (43 exemplaires étudiés) + CAROLVS REX FR croix, 4 globes AVTISIODERO CIVI temple

Chaque information pertinente est annotée. En bleu la période d'émission de la monnaie (Temporel), en vert les souverains qui l'ont faite produire (Thématique), en cyan la nature de la monnaie (Thématique) et en rouge la légende (Thématique).
Afin de développer cette chaîne,nous avons dû planifier notre travail et nos réunions avec la maîtrise d'ouvrage. Cette planification sera présentée dans la partie suivante.

III Diagramme de Gantt prévisionnel



Nom	Date de ...	Date d...
Prise de contact	04/02/15	04/02/15
Définition du projet	05/02/15	06/02/15
Documentation	06/02/15	03/03/15
Prise en Main de GATE et configurations	19/02/15	03/03/15
Formalisation du cahier de charges	03/03/15	18/03/15
Construction d'une ressource vocabulaire	03/03/15	11/03/15
Analyse du corpus	11/03/15	18/03/15
Conception d'une chaine d'annotation	19/03/15	02/04/15
Développement d'une chaine d'annotation	03/04/15	17/04/15
Mise en place interface de visualisation	20/04/15	24/04/15
Rédaction de comptes rendus	06/02/15	24/04/15
Rédaction du mémoire	03/03/15	29/04/15
Rendu du mémoire	05/05/15	05/05/15
Préparation de la soutenance	05/05/15	15/05/15
Soutenance du projet	19/05/15	19/05/15

Dans ce planning, nous avions prévu une première phase de documentation sur les différents concepts et technologies que nous allions utiliser.
En parallèle, nous devions nous mettre d'accord avec les responsables du projet sur le travail à effectuer, grâce au cahier des charges. Ensuite, nous avions la phase de développement durant laquelle nous allions mettre en place notre chaîne de traitement et notre interface de visualisation. Tout au

long du projet nous devions rédiger les différents comptes rendus de réunion et le mémoire.
Évidemment, ce diagramme ne représente en aucun cas la façon dont s'est réellement déroulé notre projet.

Troisième partie

Cadre d'analyse

Introduction

Pour cerner l'environnement du projet, nous allons définir quelques connaissances liées au domaine de l'annotation sémantique. Ensuite, viendra une description des outils qui ont été nécessaires lors de la phase de développement.

I Définition de concepts

L'annotation sémantique peut se définir comme une activité qui va mettre une "note" sur une partie d'un texte. Elle permet de travailler plus facilement sur un texte en apportant une sur-couche d'informations, qui va donner un sens aux textes.[GFAS09]

1 Le web sémantique

Le Web sémantique, ou toile sémantique, est un mouvement collaboratif mené par le World Wide Web Consortium (W3C) qui met en place des méthodes communes pour échanger des données.
Il favorise l'émergence de nouvelles connaissances en s'appuyant sur celles déjà présentes sur Internet en les structurant et en les liant.
Selon le W3C, "le Web sémantique fournit un Modèle qui permet aux données d'être partagées et réutilisées entre plusieurs applications, entreprises et groupes d'utilisateurs".
L'expression a été inventée par Tim Berners-Lee, inventeur du World Wide Web et directeur du World Wide Web Consortium qui supervise le développement des technologies communes du Web sémantique.
Le directeur du W3C définit le Web sémantique comme "un web des données qui peuvent être traitées directement et indirectement par des machines pour aider leurs utilisateurs à créer de nouvelles connaissances". [Wikb]

2 Les entités nommées

Les entités nommées sont des objets textuels (mot, ou groupe de mots) catégorisés dans des classes. Celles-ci peuvent être les noms de personnes, noms d'organisations, noms de lieux, quantités, distances, dates, auxquelles on associe un identifiant unique.[Wika]
Les entités nommées sont "associées à des expressions linguistiques sollicitées par des applications qui manipulent des documents textuels".p33 [Nov12]
Voici l'illustration d'une recherche d'entités nommées :

```
Henri a acheté 300 actions de la société AMD en 2006
<ENAMEX TYPE="PERSON">Henri</ENAMEX> a acheté
<NUMEX TYPE="QUANTITY">300</NUMEX> actions de la société
<ENAMEX TYPE="ORGANIZATION">AMD</ENAMEX> en
```

```
<TIMEX TYPE="DATE">2006</TIMEX>.
```

La reconnaissance et la résolution d'entités nommées peut s'avérer difficile dans la mesure où elles sont sujettes à des ambiguïtés liées aux phénomènes linguistiques tel que : la synonymie, l'homonymie ou encore la métonymie. p44[Nov12]

Exemple :

Avec l'entité nommée "Paris" au sein des énoncés suivants :

- "La star **Paris** Hilton s'est achetée un nouveau chien." (Personne)
- "Je suis parti en vacances à **Paris**." (Lieu)
- "Le **Paris** Saint-Germain a battu Marseille à domicile." (Organisation sportive)

3 Les gazetiers

Un gazetier peut être assimilé à un ensemble de listes contenant des noms d'entités. Elles peuvent être des villes, des organisations, des jours de la semaine, des métiers, etc. Ces listes sont utilisées pour trouver des occurrences de ces noms dans un texte, comme pour l'activité de recherche d'entités nommées.

Le terme Gazetier est souvent utilisé de manière interchangeable pour un ensemble de listes d'entités et de ressources permettant de trouver des occurrences de noms dans un texte.

Chaque gazetier est fichier texte d'extension "lst" avec une entrée par ligne.[oS]

Ci-dessous un exemple de Gazetier :

cabinet_ministers.lst ×

```
David Cameron:gender=male
Nick Clegg:gender=male
William Hague:gender=male
George Osborne:gender=male
Kenneth Clarke:gender=male
Theresa May:gender=female
Liam Fox:gender=male
Vincent Cable:gender=male
Iain Duncan Smith:gender=male
Chris Hulme:gender=male
Andrew Lansley:gender=male
Michael Gove:gender=male
Eric Pickles:gender=male
```

FIGURE 1 – Exemple de Gazetier

Sur chaque ligne, on distingue plusieurs colonnes de par des séparateurs (**:** , **;** , ...). La première colonne correspond à la valeur d'une entité, la seconde constitue le MajorType (Type principal d'annotation) de l'entité et la troisième le MinorType (Type secondaire d'annotation).[oS]

4 Les expressions régulières

Les expressions régulières sont une famille de notations compactes et puissantes pour décrire certains ensembles de chaînes de caractères. Elles permettent de rechercher automatiquement des morceaux de texte ayant certaines formes, et éventuellement remplacer ces morceaux par d'autres. Les expressions régulières sont utilisées par un grand nombre d'éditeurs de textes et utilitaires (particulièrement sous Unix), par exemple Vim, Emacs, sed ou awk. On les retrouve également dans la majorité des langages de programmation modernes.[**?**] Exemple d'utilisation d'une expression régulière sur un bout de texte :

— Expression : `(?:\d*\.)?\d+`
— Texte : **10**rats + **.36**geese = **3.14**cows

II Outils

(Il est indiqué dans notre cahier de charge, que l'annotation des textes, devra se faire dans l'environnement GATE. Il est donc impératif pour nous, de présenter cet outil.)

1 L'environnement GATE

GATE est un logiciel développé en Java à l'université de Sheffield en 1995 et utilisé pour le traitement du langage naturel, y compris l'extraction d'information dans de nombreuses langues.

Fonctionnement général

C'est un outil qui repose sur le principe d'une chaîne de traitement ("pipeline") constituée de plusieurs modules (dits "Processing ressources" PR) appliqués successivement sur un ou plusieurs textes (dits "Languages Ressources" LR). Les documents donnés en entrée peuvent être un simple texte ou un corpus en local (cf. figure 2), que l'on charge avec une URL. Les différents composants annotent chacun à leur tour le texte en prenant en compte les annotations précédentes puis le document est retourné à l'utilisateur au format XML. pp19-20[Lau10]

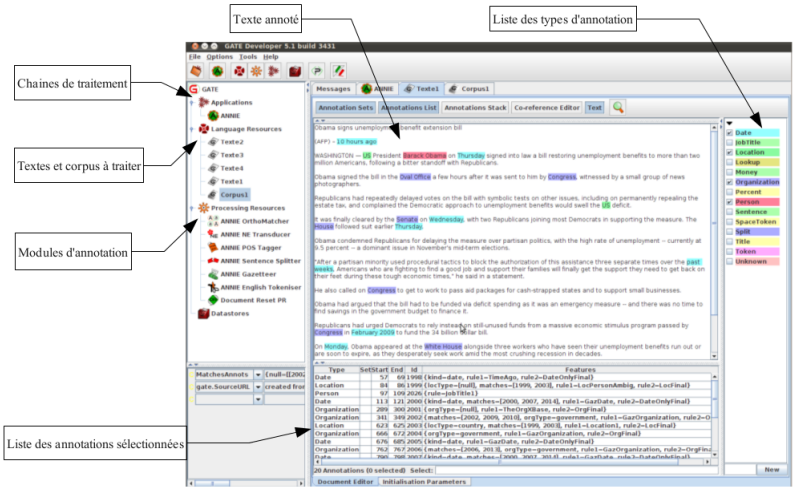


FIGURE 2 – Exemple de texte annoté dans l'outil GATE

Par un système de plugins, GATE met à disposition de ses utilisateurs un grand nombre de modules dédiés à l'analyse textuelle. Les plus courants sont les segmenteurs (Tokenizers), les étiqueteurs morpho-syntaxiques (Part Of Speech Taggers), les lexiques (Gazetteers) ou encore les transducteurs (JAPE transducers). L'interface graphique permet de charger de nouveaux plugins et ressources, de les paramétrer et de les combiner au sein d'une même chaîne de traitement (cf.figure 3). p22[Lau10]

Figure 2.5 : Exemple de chaîne de traitement GATE

FIGURE 3 – Processus d'annotation

GATE comprend entre autres un système d'extraction d'information nommé ANNIE. Il est constitué d'un certains nombre de modules, y compris ceux énoncés dans le paragraphe précédent. ANNIE est une chaîne de traitement par défaut et peut servir de point de départ pour des tâches plus spécifiques.

La chaîne de traitement ANNIE

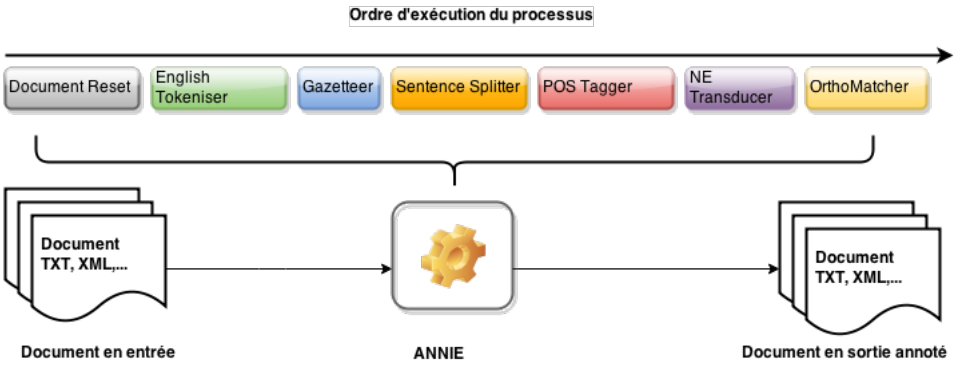


FIGURE 4 – Représentation de la chaîne de traitement ANNIE

Document Reset nettoie toutes les annotations précédentes et remet le corpus de documents à son état initial.

Tokeniser divise le texte en annotation de type Token comme par exemple les chiffres, les signes de ponctuation et des mots.

Gazetteer identifie les noms des entités qui se trouvent dans le texte, sur la base d'un gazetier.

Sentence Splitter applique une multitude de transducer à états finis qui segmente le texte en phrases.

POS Tagger attribue une catégorie linguistique à un token (Ex : NNP pour les noms propres au singulier).

NE Transducer tente de trouver les entités nommées inconnus en fonction de modèles d'extraction écrits dans le langage JAPE (expliqué dans la sous-partie suivante).

OrthoMatcher ajoute une relations d'identité entre les entités nommées trouvées par le NE Transducer, afin d'effectuer une coréférence. p24 [Lau10]

2 Le formalisme JAPE

Une partie des différents modules proposés dans GATE est basé sur le formalise JAPE (Java Annotation Patterns Engine), un transducteur à états finis

permettant de reconnaître des expressions régulières sur les annotations. Ce système s'avère très utile en extraction d'informations car il permet de définir les contextes d'apparition des éléments à extraire pour ensuite les repérer et les annoter. Le principe consiste à combiner différentes annotations dites basiques (tokens, relations syntaxiques, etc) pour en créer de nouvelles plus complexes (entités nommées, relations, événements, etc.). Cela revient à l'écriture de règles de production et donc à l'élaboration d'une grammaire régulière.
Une grammaire JAPE se décompose en plusieurs phases exécutées consécutivement et formant une cascade d'automates à états finis. Chaque phase correspond à un fichier « .jape » et peut être constituée d'une ou plusieurs règle(s) écrite(s) selon le formalisme associé à JAPE. p23[Lau10]
Classiquement, ces règles sont divisées en deux blocs :

- une partie gauche (« Left Hand Side » ou LHS) définissant un motif d'annotations à repérer,
- une partie droite (« Right Hand Side » ou RHS) contenant les opérations à effectuer sur ce motif.

Le lien entre ces deux parties se fait par l'attribution d'une étiquette au motif (ou à ses constituants) en LHS et par sa réutilisation en RHS pour y appliquer les opérations nécessaires.[Lau10]
Pour plus de clarté, prenons l'exemple d'une règle simple :
Exemple :

```
1. Rule: OrgAccronym
2. ((
3. {Organisation}
4. {Token.string == "("}
5. ({Token.orth == "allCaps"}):org
6. {Token.string == ")"}
7.  )
8.  -->
9. :org.Organisation = {rule = "OrgAcronym"}
```

L'objectif de cette règle est d'annoter en tant qu'organisation les acronymes entre parenthèses positionnés après une annotation de type "Organisation". Tout d'abord, l'on commence par donner un nom à la règle (l1). Les lignes 2 à 7 définissent le motif à repérer dans le texte. Le signe `-->` (ligne 8) sert de séparateur entre LHS et RHS. Enfin, la dernière ligne (ligne 9) exprime l'opération souhaitée.

Précisons quelques règles syntaxiques de base du formalisme JAPE :

- La partie gauche de la règle est toujours entre parenthèses
- La partie droite commence par le signe `"-->"`
- Les types d'annotation sont encadrés par des accolades

— `"Token.string"` permet d'obtenir la valeur de la propriété `"string"` associé à l'annotation `"Token"`
— `":org"` permet d'identifier une partie du motif en LHS pour l'utiliser en RHS
— La ligne 9 attribue une annotation de type "Organisation" au segment étiqueté "org" en LHS ; l'ajout de la propriété "rule" à cette annotation permet d'indiquer quelle règle en est à l'origine.

La liste des annotations utilisées en LHS de la règle est déclarée en début de phase grâce à l'attribut « Input » : par exemple, « Input : Lookup, Token, Person ». Par ailleurs, l'attribut « Control » permet de définir l'ordre d'exécution des différentes règles d'une phase et « Debug » d'obtenir un affichage des éventuels conflits rencontrés entre règles. Pour finir, précisons qu'un système de macros est également disponible : une macro permet de définir et de nommer une séquence d'annotations afin de la réutiliser plus rapidement par la suite. p23 [Lau10]
Ci-dessous un exemple de règle JAPE plus complet incluant l'utilisation des macros :
Exemple :

```
Phase: pourcent
Input: Lookup Token
Options: control = appelt
Macro: AMOUNT_NUMBER
(
({Token.kind == number}
(
({Token.string == ","} | {Token.string == "."})
{Token.kind == number}
)?
(
{Token.string == "-"}
{Token.kind == number}
)?
)
)
Macro: PERCENT
(
{Token.string == "%"} |
{Token.string == "pourcent"}|
({Token.string == "pour"}{Token.string == "cent"})
)
Macro: LIAISON_PERCENT_DE_ET
(
```

```
{Token.string == "de"}|
{Token.string == "et"}|
{Token.string == ","}
)
Rule: Percent_number
(
((AMOUNT_NUMBER)
({Token.string == "à"})
)?
(AMOUNT_NUMBER)
(PERCENT)
)
:pourcent -->
:pourcent.Percent = {Kind="pourcent",Rule="Percent_number"}
```

Pour formuler une règle JAPE, il est utile de connaître le formalisme des expressions régulières présentées en amont. Il est également utile de pouvoir tester et vérifier syntaxiquement et sémantiquement les expressions régulières réalisées.

3 RegExr : Outil en ligne pour manipuler des expressions régulières

Afin de formaliser l'écriture de nos règles d'annotations écrites dans un premier temps en langage naturel, nous avons décidé de passer par le formalisme des expressions régulières. Il nous fallait donc un outil adapté à ce travail. RegExr [2] est un outil de manipulation d'expressions régulières écrit en Javascript.
Les avantages de cet outil sont :

- la coloration syntaxique,
- la visualisation temps réel du résultat dans un texte personnalisable,
- l'aide appropriée
- une communauté partageant ses expressions régulières

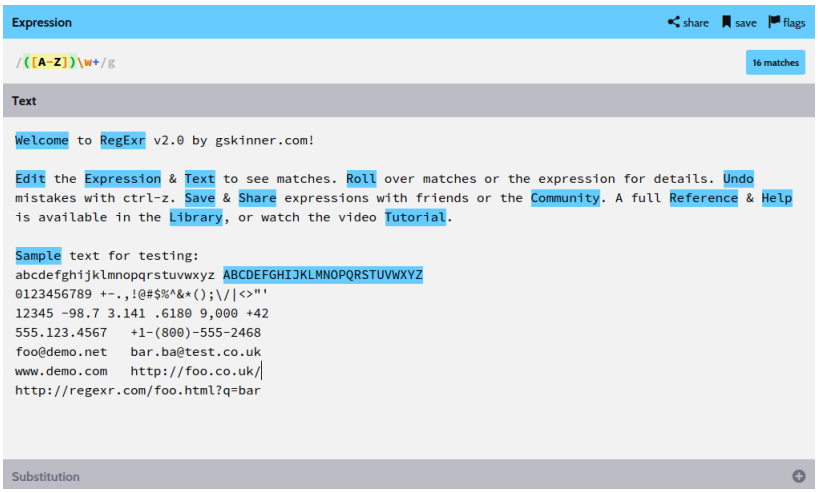


FIGURE 5 – Exemple de manipulation d'expression régulière dans RegExr

2. http ://www.regexr.com/

Quatrième partie

Développement

Introduction

Avant de démarrer le développement de la chaîne de traitement, nous devons nous familiariser avec notre environnement de travail. Ensuite, nous allons pouvoir définir nos annotations, leur domaine et choisir de quelle façon nous allons les capturer.

I Prise en main de l'environnement

Au lancement du projet, la seule ressource à disposition était le Numéraire Carolingien au format papier. Il fallait le numériser et l'OCRiser [3].
Nous avons numérisé une cinquantaine de pages à la main pour les OCRiser automatiquement par la suite à l'aide de l'outil `tesseract`.

L'OCRisation s'est déroulée de la façon suivante :

La scanner que nous avons utilisé permettait d'obtenir une image pour chacune des pages au format PDF. Ensuite, étant donné que `tesseract` est plus performant et précis avec des fichiers TIFF, il a fallu convertir les fichiers PDF en TIFF. Cependant, quelques erreurs d'OCRisation sont apparues. Pour finir, `tesseract` nous donnait des fichiers TXT.
Les étapes de conversion du schéma précédent ont été réalisées à l'aide de script en langage SHELL :

1. (a)
```
for file in *.pdf
do
  convert -density 300 ../pdf/$file
          -depth 8 ‘basename $file .pdf‘.tiff
done
```

   (b) les pages impaires ont été numérisées à l'envers, il fallait les mettre dans le bon sens.

```
for file in *.tiff
do
    if [ $((‘basename $file .tiff‘ % 2 )) = 1 ]; then
        convert $file -rotate 180 $file;
    fi
done
```

3. Reconnaissance optique de caractères

2. ```
   for file in img/*.tiff
   do
     tesseract $file txt/‘basename $file .tiff‘ -l fra
   done
   ```

II Définition des dimensions d'annotation et leur contenu

Dans l'introduction, nous avons vu qu'il y a trois dimensions d'annotation mais ne les avons pas définies.
Note : Le détail de chaque annotation peut être trouvé dans les parties III et IV.

1 Dimension spatiale

Définition

La dimension spatiale contient toutes les informations de lieux. Peu spécifique au domaine considéré.

Contenu

Le contenu de cette dimension est minime. Hormis les ateliers appartenant à la dimension thématique, les lieux présents dans le document n'intéressent que très peu la maîtrise d'ouvrage. Nous n'avons donc pas de contenu pour cette dimension.

2 Dimension temporelle

Définition

La dimension temporelle contient toutes les informations de temps et de durées. Peu spécifique au domaine considéré.

Contenu

les périodes d'émission des monnaies,

les périodes de règne des souverains.
   ```

3 Dimension thématique

Définition

La dimension thématique contient toutes les informations spécifiques au domaine considéré. C'est à dire les informations propres aux monnaies carolingiennes.

Contenu

La nature des monnaies,

La légende des monnaies

Le type des monnaies,

Les collections des monnaies,

Les trésors des monnaies,

Les trouvailles des monnaies.

III Première recherche d'entités nommées avec les gazetiers

Ateliers

■ *Vulgarisation :*

L'ouvrage est décomposé en ateliers qui donnent leur nom à chaque début de "partie" ou "chapitre". Ce nom correspond à un endroit de France ou pays limitrophes dans lequel est produite la monnaie. Ce lieu peut être une ville, un lieu-dit dont le nom peut ne plus exister. Il fût donc difficile de trouver un pattern via les expressions régulières. Nous avons alors dû construire ce gazetier à partir de la liste en début d'ouvrage qui recense tous les ateliers.
■

■ *Formalisation :*

```
Aix-la-Chapelle (Allemagne)
Agen (Lot-et-Garonne)
Aix-la-Chapelle
Alsheim (Allemagne)
Altenheim (Bas-Rhin)
Amiens (Somme)
```

■

Souverains

■ *Vulgarisation :*

Les nom des souverains ont des formats aussi divers que variés. Ils comportent des majuscules, des chiffres romains... Il serait difficile d'utiliser une expression régulière pour espérer annoter cette information. Il est plus judicieux d'utiliser un gazetier. Il sera construit à partir du début du numéraire.

■

■ *Formalisation :*

```
Pépin le Bref:valeur=Pépin le Bref:periode=752-768
Adalbert Lothaire:valeur=Adalbert Lothaire:periode=954-986
Amoul roi de Germanie:valeur=Amoul roi de Germanie:periode=887-899
Bérenger I:valeur=Bérenger I:periode=888-924
Bérenger II:valeur=Bérenger II:periode=950-961
```

■

IV Deuxième recherche d'entités nommées avec les règles JAPE

Périodes

■ *Vulgarisation :*

Période : intervalle de deux dates séparées par un tiret.
Exemple :
757/8-786

Une période est un intervalle entre deux dates. Dans notre travail, les dates sont constituées de trois et seulement trois chiffres. En cas d'ambiguïté, chacune d'entre elles peut être suivie d'un « / » et d'un chiffre traduisant l'indétermination de la date.
Exemple :
757/8

■

■ *Formalisation :*
Date

`([0-9]{3}\/?[0-9]?)`

Période

`([0-9]{3}(\/[0-9])?)-([0-9]{3}(\/[0-9])?)`

Exemple :
" *Type de 771-793/4 : Charlemagne (768-814),...*"
Group #1 : 771
Group #2 : 793/4

Group #1 : 768
Group #2 : 814

■
■ *Règle JAPE :*

```
   // Regle JAPE
  Macro: TROIS_NOMBRES
({Token.kind==number,Token.length == 3})

Macro: UN_NOMBRE
({Token.kind==number,Token.length == 1})

Macro:SLASH
({Token.string=="/"})

Macro:DATE_PRECISE
(TROIS_NOMBRES)

Macro:DATE_IMPRECISE
(TROIS_NOMBRES SLASH UN_NOMBRE)

Macro:DATE
(DATE_PRECISE | DATE_IMPRECISE)

Rule: PeriodeRule
(
  (DATE):d1({Token.string =="-"})(DATE):d2
   ):Periode -->
:Periode{/*Code java pour extraire les extremites de l'intervalle*/}
```

■

Périodes d'émission

■ *Vulgarisation :*

Une période d'émission a la forme suivante : « Type de {Période} : » (Période étant l'annotation définie précédemment).
Il faut sécuriser la capture de la période d'émission en ajoutant la contrainte précédée de "Type de" afin de ne pas récupérer toutes les périodes du document.
■
■ *Formalisation :*

`Type de : ([0-9]{3}(\/[0-9])?)-([0-9]{3}(\/[0-9])?)`

Exemple :
" Type de 771-793/4 : Charlemagne (768-814)..."
Group #1 : 771
Group #2 : 793/4

■
■ *Règle JAPE :*

```
  // Regle JAPE
  Macro: CHAINE_DEBUT
(
    ({Token.string =="Type"})({SpaceToken})
    ({Token.string =="de"})({SpaceToken})
)

Rule: PeriodeEmissionRule
(
    CHAINE_DEBUT ({Periode}):p
):PeriodeEmission
-->
:PeriodeEmission.PeriodeEmission = { Kind = "PeriodeEmission" ,D1 = :p.Periode.D1, D2 = :p.Periode.D2}
```

■

Périodes de règne

■ *Vulgarisation :*

Une période de règne a la forme suivante : « Nom_souverain ({Période}) : ».
Il faut sécuriser la capture de la période de règne en ajoutant la contrainte

précédée de "Souverain" afin de ne pas récupérer toutes les périodes du document.
Exemple :
" *Type de 771-793/4 : Charlemagne (768-814)...*"

■

■ *Formalisation :*

```
Nom_souverain ([0-9]{3}(\/[0-9])?)-([0-9]{3}(\/[0-9])?)
```

Exemple :
" *..Charlemagne (768-814)...*"
Correspondance : Charlemagne
Group #1 : 768
Group #2 : 814

■

Nature de la monnaie

■ *Vulgarisation :*

La nature de la monnaie est toujours un élément de l'ensemble {Denier, Obole, Monnaie D'Or, Faux obole, Monnaies de type indéterminé} suivi du nom d'un souverain. Il suffit donc d'utiliser une expression composée de tous les mots de l'ensemble.
Exemple :
" *Obole de Charles le Chauve*"

■

■ *Formalisation :*

```
[\s]{2,}(Denier|Obole|Monnaie d'or|Faux Obole|
Monnaies de type indéterminé)(.*)? Nom_Souverain
```

Exemple :
" *Obole de Charles le Chauve*"
Correspondance : Charles le Chauve
Group #1 : Obole

■

Légende

■ *Vulgarisation :*

La légende est toujours placée sous la ligne de la nature de la monnaie. La légende du droit est située au début de cette ligne et commence par zéro ou un caractère +. Ensuite, vient une suite d'espaces. Enfin, la légende du revers vient se placer après zéro ou un caractère +.
Exemple :
"Denier de Charlemagne .....
+ CARLO 45E CROIX SIMPLE"

■

■ *Formalisation :*

1. Se positionner à la ligne qui suit la nature de la pièce

   `(?:Denier|Obole|Monnaie d'or).*\n`

2. Capturer l'ensemble des caractères entre 0 ou 1 fois le symbole + et 2 espaces ou plus. C'est la légende du droit.

   `\+?\s?(.*)[ ]{2,}`

3. Capturer l'ensemble des caractères entre 0 ou 1 fois le symbole + et la fin de ligne. C'est la légende du revers.

   `\+?\s?(.*)\n`

On obtient une expression comme suit :

```
(?:Denier|Obole|Monnaie d'or).*\n \s*\+?\s?(.*)[ ]{2,} \+?\s?(.*)\n
```

Exemple :
"Denier de Charlemagne .....
+ CARLO 45E CROIX SIMPLE"
Group #1 : CARLO 45E
Group #2 : CROIX SIMPLE

■

Types monétaire

■ *Vulgarisation :*

Le type monétaire est la concaténation de "Type de" avec la période d'émission. Cette annotation est similaire à la période d'émission mais appartient à la dimension thématique.
Exemple :
" Type de 771-793/4 : Charlemagne (768-814),..."

■
■ *Formalisation :*

```
(Type de [0-9]{3}\/?[0-9]?-[0-9]{3}\/?[0-9]?)
```

Exemple :
"Type de 771-793/4 : Charlemagne (768-814),...
Group #1 : Type de 771-793/4

■

Collections, Trésors, Trouvailles

■ *Vulgarisation :*

Les collections, trésors et trouvailles sont chacun des ensembles d'informations à annoter séparément. Les *mots* Collections, Trésors et Trouvailles sont chacun suivis de deux points. Ensuite, vient le contenu concernant ces mots. Le contenu s'arrête lorsqu'on rencontre un point suivi d'un retour à la ligne ou bien un autre *mot* suivi de deux points.
■
■ *Formalisation :*

```
Mot_a_trouver:((?:.|\n)+?)(?:\.\s?\n|\w:)
```

Exemple :
"...Collections : Berlin 1,77, 1,70, 1,59, 1,55 ; MEC 853 (1,78) ; Monnaie de Paris 105 (1,63) ; Prou 584 (1,58), 585 (1,69), 586 (1,79), 587 (1,72), 588 (1,77). Trésors :..."
Group #1 : Berlin 1,77, 1,70, 1,59, 1,55 ; MEC 853 (1,78) ; Monnaie de Paris 105 (1,63) ; Prou 584 (1,58), 585 (1,69), 586 (1,79), 587 (1,72), 588 (1,77).
Trésors :

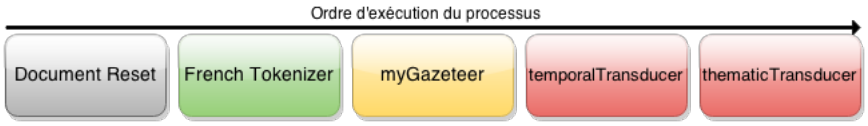
■

Conclusion

Toutes les règles JAPE sont faites, il faut maintenant les placer dans le bon ordre. C'est à dire, mettre en premier les règles qui ne dépendent d'aucune autre puis les règles restantes. Une fois fait, on est à même d'annoter l'ensemble d'informations voulu grâce à notre chaîne de traitement présentée dans la partie suivante.

V Bilan du développement

Nous arrivons aux termes de notre développement. Deux gazetiers ont été crées et une douzaine de règles JAPE.
Nous sommes partis de la chaîne de traitement ANNIE. Nous avons retiré les modules qui ne nous étaient pas utiles et ajouté les nôtres. On arrive à la chaîne suivante :



N'ayant besoin que de repérer les mots (Token) pour notre annotation, nous n'avons gardé que le Tokenizer. Ensuite,

myGazeteer contient les deux gazetiers que nous avons réalisés.

temporalTransducer contient les règles JAPE du domaine Temporel.

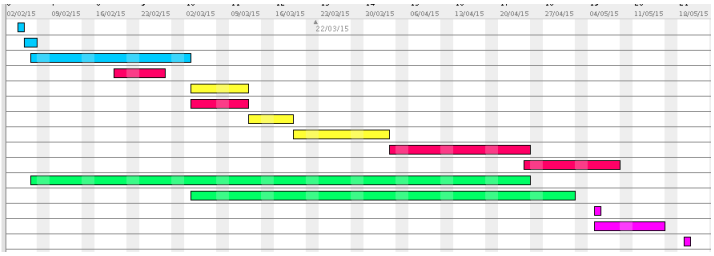
thematicTransducer contient les règles JAPE du domaine Thématique.

Cinquième partie

Conclusion

I Bilan du projet

Nom	Date d...	Date d...
Prise de contact	04/02/15	04/02/15
Définition du projet	05/02/15	06/02/15
Documentation	06/02/15	02/03/15
Prise en main de GATE et configurations	19/02/15	26/02/15
Formalisation du cahier de charges	03/03/15	11/03/15
Construction d'une ressource vocabulaire	03/03/15	11/03/15
Analyse du corpus	12/03/15	18/03/15
Conception d'une chaîne d'annotation	19/03/15	02/04/15
Développement d'une chaîne d'annotation	03/04/15	24/04/15
Mise en place interface de visualisation	24/04/15	08/05/15
Redaction de comptes rendus	06/02/15	24/04/15
Rédaction du mémoire	03/03/15	01/05/15
Rendu du mémoire	05/05/15	05/05/15
Préparation de la soutenance	05/05/15	15/05/15
Soutenance du projet	19/05/15	19/05/15



En comparant notre planning effectif au planning prévisionnel, il ressort que nous avons respecté le planning prévisionnel pour la première phase du projet et aussi toute la phase de rédaction des différents comptes rendus de réunion. La première phase comprenant la documentation, la prise en main des outils de travail, la formalisation du cahier de charge et l'étape de conception. Ce fut une phase de recherche pendant laquelle il nous est paru nécessaire de de bien cerner le domaine d'étude et aussi de mettre en place une bonne conception. Nous avions donc prévu pour cette phase une période de travail plus longue. En revanche, nous fut impossible de respecter le planning prévisionnel pour la phase de développement. En effet, notre chaîne de traitement est aboutie mais l'interface web de visualisation des résultats d'annotation quant à elle est toujours en cours de développement et sera présentée lors de la soutenance du projet.

II Expériences acquises

Il est important de noter plusieurs retours d'expérience.
D'abord la gestion du temps pour suivre convenablement un planning prévisionnel : nous retenons qu'il est assez improbable de respecter un planning prévisionnel, surtout pendant la phase de développement en raison du fait qu'on peut être confronté à des difficultés techniques (code) dans cette étape.
Ensuite la capacité de pouvoir fournir un produit final à un client qui respecte au mieux les spécifications exprimées dans un cahier de charges. On peut noter qu'il est primordial d'être fréquemment en contact avec le client pour mieux cerner et comprendre ses besoins. En effet, sur ce projet, le rôle de client fût joué par nos encadrants et celui du prestataire joué par nous. Enfin, la préparation des réunions de travail qui passe par la rédaction d'ordres du jour et de comptes rendus, est une expérience supplémentaire que nous avons acquise.
Nous pensons que toutes ces expériences pourront nous être utiles à l'avenir dans le monde professionnel.

Bibliographie

[GFAS09] A. Guissé, F.Lévy, A.Nazarenko, and S.Szulman. Annotation sémantique pour l'indexation de règles métiers, 2009.

[Lau10] SERRANO Laurie. Modélisation d'une ontologie de domaine et des outils d'extraction de l'information associés pour l'anglais et le français, 2010.

[Nov12] Damien Novel. Reconnaissance des entitées nommées par exploration de règles d'annotation, 2012.

[oS] The University of Sheffield. Gazetteers. `https://gate.ac.uk/sale/tao/splitch13.html`.

[Wika] Wikipedia. Reconnaissance d'entités nommées. `http://fr.wikipedia.org/wiki/Reconnaissance_d%27entit%C3%A9s_nomm%C3%A9es`.

[Wikb] Wikipedia. Web sémantique. `http://fr.wikipedia.org/wiki/Web_s%C3%A9mantique`.

Annexe A

Règles JAPE

periode.jape

```
Phase: periode
Input: Token
Options: control = appelt

Macro: TROIS_NOMBRES
({Token.kind==number,Token.length == 3})

Macro: UN_NOMBRE
({Token.kind==number,Token.length == 1})

Macro:SLASH
({Token.string=="/"})

Macro:DATE_PRECISE
(TROIS_NOMBRES)

Macro:DATE_IMPRECISE
(TROIS_NOMBRES SLASH UN_NOMBRE)

Macro:DATE
(DATE_PRECISE | DATE_IMPRECISE)

/**
* Description : Capture une période
*
* Exemple : 758-542 , 758/9-762, ...
*/
Rule: PeriodeRule
(
  (DATE):d1({Token.string =="-"})(DATE):d2
    ):Periode -->
:Periode{
     //En entree
 OffsetComparator comparator = new OffsetComparator();
 gate.AnnotationSet matchedD1 = (gate.AnnotationSet)bindings.get("d1");
 gate.AnnotationSet matchedD2 = (gate.AnnotationSet)bindings.get("d2");
 List<Annotation> listD1 = new ArrayList<Annotation>(matchedD1);
 List<Annotation> listD2 = new ArrayList<Annotation>(matchedD2);
 Collections.sort(listD1, comparator);
 Collections.sort(listD2, comparator);
```

```
 String date_1 = "";
 String date_2 = "";

    //On récupere toute la chaine D1, while necessaire pour récuperer les /
 for (Annotation ann : listD1) {
  date_1 += (String)ann.getFeatures().get("string");
}
for (Annotation ann : listD2) {
  date_2 += (String)ann.getFeatures().get("string");
}

      // Traitement des deux chaines. on prend le min si D1 est imprecise et le max si d2 est imprécise
if (date_1.contains("/")) {
  date_1 = date_1.substring(0, date_1.indexOf("/"));
}
if (date_2.contains("/")) {
  date_2 = date_2.substring(0, date_2.indexOf("/")-1) + date_2.substring(date_2.indexOf("/") + 1);
}

      //declaration d'annotation vide
gate.AnnotationSet matchedInter = (gate.AnnotationSet)bindings.get("Periode");
gate.FeatureMap newFeatures = Factory.newFeatureMap();
      //MAJ de l'annotation initialement vide
newFeatures.put("Kind","Periode");
newFeatures.put("Rule","PeriodeRule");
newFeatures.put("D1",date_1);
newFeatures.put("D2",date_2);

      //Validation de l'annotation et sortie
outputAS.add(matchedInter.firstNode(),matchedInter.lastNode(),"Periode",newFeatures);
}
```

periode_emission.jape

```
Phase: periode_emission
Input: Token Periode SpaceToken
Options: control = appelt

Macro: CHAINE_DEBUT
(
    ({Token.string =="Type"})({SpaceToken})
    ({Token.string =="de"})({SpaceToken})
)


/**
* Description : Capture une période d'emission
*
* Exemple : Type de 758/9-762
*/
Rule: PeriodeEmissionRule
(
    CHAINE_DEBUT ({Periode}):p
):PeriodeEmission
-->
:PeriodeEmission.PeriodeEmission = { Kind = "PeriodeEmission" ,D1 = :p.Periode.D1, D2 = :p.Periode.D2}
```

legende.jape

```
Phase: legende
Input: Token SpaceToken NatureMonnaie
```

```
Options: control = appelt


Macro:FIN_LIGNE
(
    {SpaceToken.kind==control, SpaceToken.string !=~ "[\\t]"}
)

Macro:EXEMPLAIRES
(
((({Token})|({SpaceToken.kind == space}))*)
)

Macro: LIGNE_PRECEDENTE
 (({NatureMonnaie})(EXEMPLAIRES)?(FIN_LIGNE))

Macro: LIAISON
({Token.string == "+"})

Macro: ESPACEMENT
(({SpaceToken})({SpaceToken})({SpaceToken})*)

Macro: LEGENDE
(
((({Token})({SpaceToken})?)*)
)

//Règles
Rule: LegendeRule
(LIGNE_PRECEDENTE)
(
    (LIAISON)?({SpaceToken})*(LEGENDE):droit(ESPACEMENT)(LIAISON)?({SpaceToken})*(LEGENDE):revers
):Legende
-->
        :Legende{
            //En entree
 OffsetComparator comparator = new OffsetComparator();
 gate.AnnotationSet matchedDroit = (gate.AnnotationSet)bindings.get("droit");
 gate.AnnotationSet matchedRevers = (gate.AnnotationSet)bindings.get("revers");
 List<Annotation> listDroit = new ArrayList<Annotation>(matchedDroit);
 List<Annotation> listRevers = new ArrayList<Annotation>(matchedRevers);
 Collections.sort(listDroit, comparator);
 Collections.sort(listRevers, comparator);

 String legende_droit = "";
 String legendre_revers = "";

//On récupere chaque token séparés par les expaces
 for (Annotation ann : listDroit) {
  legende_droit += (String)ann.getFeatures().get("string");
}
for (Annotation ann : listRevers) {
  legendre_revers += (String)ann.getFeatures().get("string");
}

           //declaration d'annotation vide
gate.AnnotationSet matchedInter = (gate.AnnotationSet)bindings.get("Legende");
gate.FeatureMap newFeatures = Factory.newFeatureMap();
           //MAJ de l'annotation initialement vide
newFeatures.put("Kind","Legende");
newFeatures.put("Rule","LegendeRule");
newFeatures.put("Droit",legende_droit);
```

```
newFeatures.put("Revers",legendre_revers);

          //Validation de l'annotation et sortie
outputAS.add(matchedInter.firstNode(),matchedInter.lastNode(),"Legende",newFeatures);
       }
```

nature_monnaie.jape

```
/*  nature_monnaie.jape
* annotation de la nature des monnaies
* utilisation d'expressions régulières dans une règle JAPE
* simple règles JAPE
* Roland Bary - Charles Follet
* Mars 2015
*/
Phase: nature_monnaie
Input: Lookup Token SpaceToken ANgazetier
Options: control = appelt


Macro: LIAISON
({Token.string == "de"})

Macro: MONNAIE
(
  ({Token.string == "Obole"})|
  ({Token.string == "Denier"})|
  ({Token.string == "Monnaie d'or"})|
  ({Token.string == "Faux Obole"})|
    ({Token.string == "Monnaies de type indéterminé"})
)


/**
* Description : Capture la nature de la monnaie
*
* Exemple : Denier de Charlemagne
*/
Rule: NatureMonnaieRule
(
  ({SpaceToken})?(MONNAIE):monnaie({SpaceToken})(LIAISON)({SpaceToken})({ANgazetier.majorType == "rois"}):roi

):NatureMonnaie
-->
    :NatureMonnaie.NatureMonnaie = {Kind = "NatureMonnaie", Monnaie = :monnaie.Token.string, Roi = :roi.ANgazetie
```

tresors.jape

```
Phase: Tresors
Input: Lookup Token SpaceToken
Options: control = appelt

Macro: SEPARATOR
(
    ({Token.string == ":"})
)

Macro: TRESORS
(
       ({Token.string == "Trésors"})(SEPARATOR)({SpaceToken})
)
```

```
/**
* Description : Capture le mot clef tresors dans le texte (utile pour ensuite délimiter le contenu de tresors)
*
* Exemple : Trésors:
*/
Rule: TresorsRule
(
    (TRESORS):tresors
):Tresors
-->
       :Tresors{
 //En entree
 OffsetComparator comparator        = new OffsetComparator();

 gate.AnnotationSet matchedTresors = (gate.AnnotationSet)bindings.get("tresors");
 List<Annotation> listTresors    = new ArrayList<Annotation>(matchedTresors);

 Collections.sort(listTresors, comparator);

 String tresors = "";

//On récupere chaque token séparés par les expaces
 for (Annotation ann : listTresors) {
  tresors += (String)ann.getFeatures().get("string");
}

//            //declaration d'annotation vide
 gate.AnnotationSet matchedInter = (gate.AnnotationSet)bindings.get("Tresors");
 gate.FeatureMap newFeatures     = Factory.newFeatureMap();
//            //MAJ de l'annotation initialement vide
newFeatures.put("Kind","Tresors");
newFeatures.put("Rule","TresorsRule");
newFeatures.put("Tresors",tresors);

          //Validation de l'annotation et sortie
outputAS.add(matchedInter.firstNode(),matchedInter.lastNode(),"Tresors",newFeatures);
       }
```

tresors_details.jape

```
Phase: tresors_details
Input: Token Collections Tresors Trouvailles PeriodeEmission NatureMonnaie ANgazetier
Options: control = appelt

/**
* Description : Capture le contenu de trésors
*
* Exemple : Trésors: Lorem ipsum Exercitation fugiat nulla nulla.
*/
Rule: TresorsDetails
(
    ({Tresors})
    (({Token,!Collections,!Trouvailles,!PeriodeEmission,!NatureMonnaie,! ANgazetier.majorType == "ateliers"})+):ad
    ({Collections}|{Trouvailles}|{ANgazetier.majorType == "ateliers"}|{PeriodeEmission}|{Token.kind =="number"}{NA
    )
-->
: ad{
       //En entree
   OffsetComparator comparator        = new OffsetComparator();
```

```
   gate.AnnotationSet matchedTresors = (gate.AnnotationSet)bindings.get("ad");
   List<Annotation> listCoTresors    = new ArrayList<Annotation>(matchedTresors);

   Collections.sort(listCoTresors, comparator);

   String tresors = "";

           //On récupere chaque token séparés par les expaces
   for (Annotation ann : listCoTresors) {
      tresors += (String)ann.getFeatures().get("string");
  }

           //declaration d'annotation vide
  gate.AnnotationSet matchedInter = (gate.AnnotationSet)bindings.get("ad");
  gate.FeatureMap newFeatures     = Factory.newFeatureMap();
           //MAJ de l'annotation initialement vide
  newFeatures.put("Kind","tresors");
  newFeatures.put("Rule","TresorsDetails");
  newFeatures.put("Details",tresors);

           //Validation de l'annotation et sortie
  outputAS.add(matchedInter.firstNode(),matchedInter.lastNode(),"TresorsDetails",newFeatures);
}
```

trouvailles.jape

```
Phase: Trouvailles
Input: Lookup Token SpaceToken
Options: control = appelt

Macro: SEPARATOR
(
    ({Token.string == ":"})
)

Macro: Trouvailles
(
       ({Token.string == "Trouvailles"})(SEPARATOR)({SpaceToken})
)

/**
* Description : Capture le mot clef trouvaille dans le texte (utile pour ensuite délimiter le contenu de la trouv
*
* Exemple : Trouvailles:
*/
Rule: TrouvaillesRule
(
    (Trouvailles):trouvailles
):Trouvailles
-->
       :Trouvailles{
 //En entree
 OffsetComparator comparator        = new OffsetComparator();

 gate.AnnotationSet matchedTrouvailles = (gate.AnnotationSet)bindings.get("trouvailles");
 List<Annotation> listTrouvailles   = new ArrayList<Annotation>(matchedTrouvailles);

 Collections.sort(listTrouvailles, comparator);

 String trouvailles = "";
```

```
//On récupere chaque token séparés par les expaces
 for (Annotation ann : listTrouvailles) {
  trouvailles += (String)ann.getFeatures().get("string");
}

//            //declaration d'annotation vide
 gate.AnnotationSet matchedInter = (gate.AnnotationSet)bindings.get("Trouvailles");
 gate.FeatureMap newFeatures     = Factory.newFeatureMap();
//            //MAJ de l'annotation initialement vide
newFeatures.put("Kind","Trouvailles");
newFeatures.put("Rule","TrouvaillesRule");
newFeatures.put("Trouvailles",trouvailles);

           //Validation de l'annotation et sortie
outputAS.add(matchedInter.firstNode(),matchedInter.lastNode(),"Trouvailles",newFeatures);
       }
```

trouvailles_details.jape

```
Phase: tresors_details
Input: Token Collections Tresors Trouvailles PeriodeEmission NatureMonnaie ANgazetier
Options: control = appelt

/**
* Description : Capture le mot contenu de trouvailles
*
* Exemple : Trouvailles: Lorem ipsum Enim cupidatat Excepteur.
*/
Rule: TrouvaillesDetails
(
    ({Trouvailles})
    (({Token,!Collections,!Tresors,!PeriodeEmission,!NatureMonnaie,! ANgazetier.majorType == "ateliers"})+):ad
    ({Collections}|{Tresors}|{ANgazetier.majorType == "ateliers"}|{PeriodeEmission}|{Token.kind =="number"}{Natur
    )
-->
: ad{
       //En entree
   OffsetComparator comparator        = new OffsetComparator();

   gate.AnnotationSet matchedTresors  = (gate.AnnotationSet)bindings.get("ad");
   List<Annotation> listCoTresors     = new ArrayList<Annotation>(matchedTresors);

   Collections.sort(listCoTresors, comparator);

   String trouvailles = "";

           //On récupere chaque token séparés par les expaces
   for (Annotation ann : listCoTresors) {
      trouvailles += (String)ann.getFeatures().get("string");
  }

           //declaration d'annotation vide
  gate.AnnotationSet matchedInter = (gate.AnnotationSet)bindings.get("ad");
  gate.FeatureMap newFeatures     = Factory.newFeatureMap();
           //MAJ de l'annotation initialement vide
  newFeatures.put("Kind","trouvailles");
  newFeatures.put("Rule","TrouvaillesDetails");
  newFeatures.put("Details",trouvailles);

           //Validation de l'annotation et sortie
  outputAS.add(matchedInter.firstNode(),matchedInter.lastNode(),"TrouvaillesDetails",newFeatures);
```

```
}
```