General Architecture for Text Engineering - GATE

basierend auf dem GATE-Benutzerhandbuch sowie
dem Tutorial des CLab-Teams der Universität Zürich

3. Juni 2011

1 Lernziele

- Grundkenntnisse in GATE Development Environment erlangen
- Erfahrungen sammeln mit einer Annotationsoberfläche zur Informationsextraktion
- Einfache Modifikationen an JAPE Grammatiken zur Erkennung von NER
- Entwicklung einer eigener einfachen NLP Applikation mithilfe der GATE Development Environment.

2 Installation

Lade Dir auf der Seite `http://gate.ac.uk/download/` das binary-only Package herunter. Lade die aktuelle release runter und keine Beta-Version! GATE 6.1 ist momentan die aktuelle Version. GATE 6.1 benötigt eine mindestens Java-5.0-fähige Umgebung. Falls Du das Java SE Development Kit nicht installiert hast, musst Du das vor GATE Installation machen.

ACHTUNG: Falls Du ein 64bit Windows hast, nehme die 32bit-JDK!

Entpacke die gezippte Datei. Starte nun GATE mit `gate.exe` für Windows, GATE.app für Mac und für Linux mit dem Skript gate.sh im Unterordner /bin .

3 Was ist GATE

GATE ist eines der bekanntesten Sprachverarbeitungssysteme mit integrierten Stufen der automatischen Sprachverarbeitung wie z.B. Tokenisierung, Wortarten-Bestimmung, Named-Entity-Recognition, Koreferenzauflösung usw.

GATE beinhaltet drei Komponenten:

1. GATE Document Manager (GDM): Schnittstelle zum Speichern von Informationen über Texte die eine objektorientierte Manipulierung der Datenbank zulässt

2. GATE Graphical Interface (GGI): Graphische Oberfläche des Systems, die es dem Benutzer erlaubt die Daten und Module zu überwachen, auszuwerten und zu manipulieren
3. Collection of Reusable Objects for Language Engineering (CREOLE): Beinhaltet Module wie Tokeniser, Tagger, Parser, welche die linguistische Verarbeitung als einheitliches System durchführt

Bei GATE Komponenten handelt es sich um Java Beans die in drei Hauptkategorien aufgeteilt sind:

1. Language Resources (LR): Korpora, Lexika, Ontologien
2. Processing Resources (PR): Programme oder Algorithmen die den Text bearbeiten z.B Parser, POS, Tokeniser etc.
3. Visual Resources (VR): Visualisierung und Bearbeitung der in der GUI vorhandenen Komponenten

Folgende Dokumentformate sind als Input zugelassen: einfache Text-Dokumente, HTML, XML, SGML, RTF, Email und einige PDF und Microsoft Word Dokumente. Die bearbeiteten Dateien können als XML-Datei abgespeichert werden.

3.1 Korpus erstellen

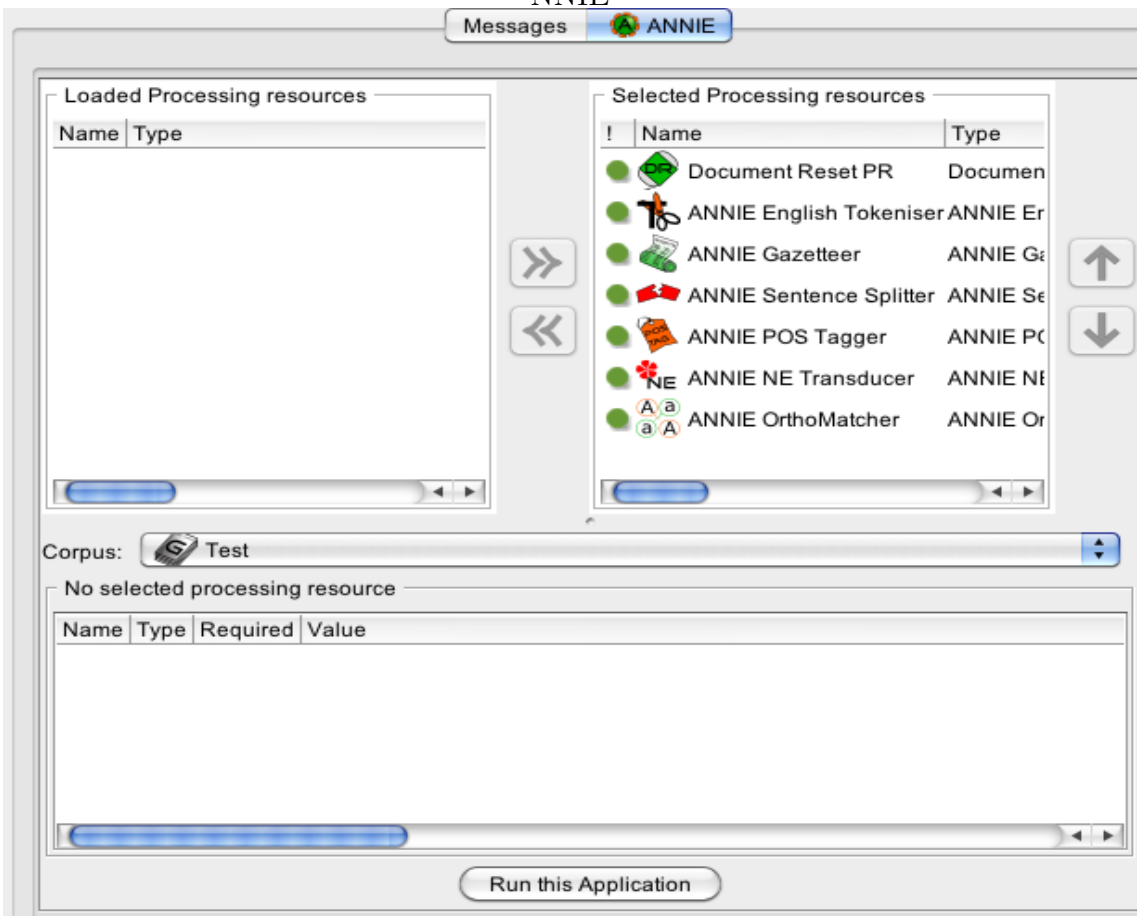
Erstelle als Erstes aus den Dir zur Verfügung gestellten Texten einen Korpus. Hierzu kannst Du Dir die Instruktionsfilme Module 1-3 auf `http://gate.ac.uk/demos/developer-videos/` anschauen. Die Codierung der Dokumente ist UTF-8.

3.2 ANNIE

ANNIE (A Nearly-New Information Extraction system) ist das in GATE implementiertes IE-System. Für ausführliche Informationen siehe Kapitel 6 in GATE User Guide `http://gate.ac.uk/sale/tao/index.html#x1-1260006`.

Nachdem Du den Korpus erstellt hast, kannst Du nun ANNIE über die Texte laufen lassen. Klicke auf das ANNIE-Zeichen und wähle `with Defaults`. Per Doppelklick auf ANNIE unter Applikationen kannst Du rechts die ANNIE-Verarbeitungskette sehen. Die Reihenfolge der Plugins ist wichtig (Abbildung 1).

Abbildung 1: A
NNIE



3.2.1 Gazatteer

Klicke nun auf den Text `SDA.941104.0166_00025` und schaue Dir die markierten Entitäten für `Date` an. Es werden nur die Entitäten gefunden, die auch in ANNIE Gazatteer vorkommen.

Das Processing Ressource (das Plugin) Gazetteer ist sehr simpel und besteht aus Listen für verschiedene Bezeichnungen, zum Beispiel Namen von Personen, Organisationen, Datum, Jobtiteln etc. Sie funktioniert über Gazetteerlisten. Die Gazetteerlisten der Applikation ANNIE sind unter `GATE-6.1/plugins/ANNIE/resources/gazetteer/` zu finden. Die Listen sind einfache Textdokumente und enden auf `.lst`. Pro Zeile ist ein Eintrag vorhanden z.B.:

aqua
beige
black
...
Das Plugin benutzt eine Indexdatei, die besagt, welche Gazetteerlisten abgerufen werden sollen (`GATE-6.1/plugins/ANNIE/resources/gazetteerlists.def`). Der Index muss sich im selben Verzeichnis wie die einzelnen Listen befinden. Die Listen können erweitert bzw. modifiziert werden. Die Listen können entweder direkt in GATE oder in einem externen Editor bearbeitet werden.

3.2.2 JAPE - a Jolly And Pleasant Experience

Java Annotation Pattern Engine ist ein regelbasierter Formalismus in GATE und enthält viele LHS (left hand side) und RHS (right hand side) Regeln, die wie folgt aufgebaut sind:

```
{ Regel: Name Muster --> Aktion}
```

Links vor dem Pfeil steht die LHS-Regel für das gesuchte Muster und rechts vor dem Pfeil beschreibt die RHS-Regel die Aktion, die ausgeführt werden soll. Die LHS-Regeln benutzen reguläre Operationen (—, ?, *, +), während die RHS-Regeln Blöcken von Java Codes zu Verfügung haben, mit denen sie die Annotationen manipulieren. Ausführlichere Informationen über Anwendung und Modifizierung von JAPE findest Du auf `http://gate.ac.uk/sale/tao/splitch8.html#chap:jape`.

In der Datei `Adressen.txt` sind einige Adressen aufgelistet. Wenn Du nun ANNIE über dieses Dokument laufen läst, erkennt der NE-Transductor die meisten `Locations` mit englischen Adressen, die deutsche Adresse wie `Baumschuhl Allee` dagegen nicht. Das liegt natürlich daran, dass hier die Regel zum Erkennen der Adresse auf englischen Tokens basiert ist, und muss in unseren Fall erweitert werden. Schau Dir die Regel mit dessen Hilfe Gazatteer das Tag `Adresse` erstellt. Die Datei findest Du unter `GATE-6.1/plugins/ANNIE/resources/NE/adress.jape`. Du kannst auch GATEs eigenen UTF-8 Editor benutzen (Menüleiste 'tools'). Auch hier verwende bitte UTF-8 Codierung und kein 'unicode'.

Schaue Dir dazu nun die Regel `StreetName1` an. Wie bereits erwähnt ist die linke Seite der Regel für das Matchen zuständig. Gesucht wird also ein Token, das einer Zahl ist. Diese kann von einem optionalen Komma gefolgt von einem Token mit Großbuchstabe, gefolgt von Token aus Street-Gazatteer sein.

```
Rule: StreetName1
(
({Token.kind == number}
({Token.string == ","})?
)?
{Token.orth == upperInitial}
{Lookup.minorType == "street"}
 )
 :streetAddress
```

Ist das gesuchte Muster gefunden worden, führt die rechte Seite der Regel die gewünschte Aktion aus. In unserem Fall heißt das, dem gefundenen Token wird ein Label `streetAdress` zugewiesen, dem eine Annotation (Markierung) vom Typ `Street` zugeordnet wird. Der Regelname `StreetName1` wird als Attribut der Annotation gesetzt, um nachzuverfolgen, welche Regel für welche Annotation zuständig ist.

Nun musst Du das Token 'Allee' zu den Einträgen der `street.lst` hinzufügen. In der Datei stehen die von der Gazatteer `street` benötigten Informationen, nämlich Straßennamen.
Öffne die Datei `street.lst` und füge das Token 'Allee' hinzu. Lasse nun ANNIE nochmal über die Texte laufen, nachdem Du deine Änderungen abgespeichert hast. Wenn Du die gesamte Adresse samt Nummer annotieren möchtest, musst Du die JAPE-Regeln in der Datei `adress.jape` noch wie folgt erweitern:

```
Rule: StreetName2
 (
```

```
{Token.orth == upperInitial}
{Lookup.minorType == "street"}
{Token.kind == number}
 )
:streetAddress -->
 :streetAddress.Street = {kind = "streetAddress", rule = StreetName2}
```

Lade per Rechtsklick nun den modifizierten NE Transducer neu ('reinitialize'). Wenn Du nun ANNIE nochmal laufen lässt, muss auch die deutsche Adresse `Baumschuhl Allee 22` als Straße erkannt werden.

4 Abspeichern von Dateien und Applikationen

Nach jeder Bearbeitung musst Du natürlich deine Ergebnisse abspeichern, sonst sind diese bei nächstem Neustart nicht mehr vorhanden. Klick links in der GUI auf `Data Store` und wähle `Create datastore`. Hier kannst Du deinen Korpus samt allen von Dir annotierten Dokumenten mit `Save to Datastore` für spätere Verarbeitungen speichern.

In dem Instruktionsfilm Module 7 auf `http://gate.ac.uk/demos/developer-videos/` findest Du die genaue Vorgehensweise.

Du kannst in GATE auch Deine eigene Applikationspipeline bauen. Klick links in der GUI auf `Applications` und wähle `Corpus pipeline` (siehe Instruktionsfilm Module 5). Auf diese Weise kannst Du Deinen Korpus, Deine Dokumente und die von Dir zusätzlich benutzte Processing Resources in Deiner Pipeline samt ihren Laufzeitparametern abspeichern (z.B. einen von Dir geschriebenen Jape Transducer). Diese gespeicherte `.gapp` Datei ist auch das Abgabeformat für diesen Block.

4.1 Weitere Informationen

Für eine detaillierte Einführung in GATE, schaue Dir das GATE-Benutzerhandbuch unter `http://gate.ac.uk/sale/tao/split.html.` an.