Using GATE as an Annotation Tool

Tom Kenter, Diana Maynard

20th July 2004

Contents

1 Introduction

This manual is designed as an introduction to GATE 2 for people who have no experience at all with the tool. The first part covers the basic aspects of how to use GATE as an annotation tool; the second part includes some more advanced aspects concerned with using the ontology functionalities. For more detailed information, we refer the reader to the GATE User Guide (see Section 3.2). Note that some parts of the GUI have changed significantly in Version 3 of GATE. If you are using GATE 3, please refer to the corresponding documentation for GATE 3.

1.1 What is GATE for?

But first, what is GATE for?

GATE can be used for infinitely many things, but one of the most typical uses is to annotate pages with it. This means that you have a collection of pages (a corpus) and a number of concepts (Annotation Schema) that supposedly occur in these pages. GATE provides you with an easy to use interface for indicating which pieces of text denote which of your concepts. In GATE you can do the annotating by hand, or you can let GATE do this automatically by using Gazetteers, etc. For example, GATE automatically annotates all html tags it finds in your text (you will find them in the Annotation Set called 'Original markups annotations').

1.2 Overview of GATE

GATE is an architecture that contains functionality for plugging in all kinds of NLP software, such as POS taggers, sentence splitters, Named Entity recognizers, etc. It works with resources. There are two main kinds of resources: Language Resources, and Processing Resources.

- Language Resource (LR): refers to data-only resources such as lexicons, corpora, thesauri or ontologies. Some LRs come with software (e.g. Wordnet has both a user query interface and C and Prolog APIs), but where this is only a means of accessing the underlying data we will still define such resources as LRs.
- Processing Resource (PR): refers to resources whose character is principally programmatic or algorithmic, such as lemmatisers, generators, translators, parsers or speech recognisers. For example, a part-of-speech tagger is best characterised by reference to the process it performs on text. PRs typically include LRs, e.g. a tagger often has a lexicon; a word sense disambiguator uses a dictionary or thesaurus.

These resources can be loaded into GATE, and saved together in a Data Store. Also, Processing Resources can be put together in a so-called 'pipeline' (as in Unix pipelines, where the output of an application serves as input to the next). This is called an 'application' in GATE.

2 Getting started

2.1 Download and install the software

You can download the GATE annotation tool from: http://gate.ac.uk/download/index.html

You will be asked for some details, and download instructions will be sent to you by mail. Once you have downloaded the application, you can install it in a place convenient for you.

2.2 Documentation

By default, GATE comes with documentation included. The main source of information is the User Guide ('Developing Language Processing Components with GATE (a User Guide)') which is called tao.pdf, located in the directory that GATE was installed in. Or you can view it online at http://gate.ac.uk/sale/tao

Also there is the very useful mailing list, and accompanying archive. Information about subscribing to the list, and browsing the mailing list archive, can be found at http://gate.ac.uk/mail/index.html.

2.3 A GATE session

2.3.1 Start the GATE application

Click on the icon if you choose to have one installed, or double click on the gate.bat file in GATE's bin/ directory.

2.3.2 Importing/loading/saving resources

Throughout the documentation about GATE, importing/opening files is called 'loading'. We will adhere to this convention.

You can load both Language Resources and Processing Resources, by right clicking on 'Language/Processing Resource' in GATE's left window, and choosing 'New'. Or by going to the 'File' menu and choosing 'New Language/Processing Resource'.

NOTE that 'new' is being used here as 'open', ie it is used for opening existing files, rather than for creating completely new ones. The 'new' is local, i.e. it means 'new to the particular application' rather than globally new.

By default GATE loads no resources. However the tool can be configured to save session data on closing. In this case, all the data will be loaded again automatically the next time it is run. This is done by choosing 'Configuration' in the 'Options' menu. Under the 'Advanced options' it can be specified whether or not session info and/or options should be saved upon exitting. Saving options just stores the settings such as fonts etc, rather than the data.

2.3.3 Annotation schemas

Annotation schemas provide a means to define types of annotations in GATE - basically this means that GATE "knows about" annotations defined in a schema. The default annotation schema contains common named entities such as Person, Organisation, Location, etc. You can modify the existing schema or create a new one, in order to tell GATE about other kinds of annotations you frequently use. You can still create annotations in GATE without having specified them in an annotation schema, but you may then need to tell GATE about the properties of that annotation type each time you create an annotation for it. Section 5.4.1 "Annotation Schemas" in the GATE User Guide describes how to create new schemas.

2.3.4 Start annotating

Load the file that is to be annotated.

To open a file, select 'New Language Resource' in either of the two menus, and then 'GATE document'. A window will appear in which the parameters for the document can be stated. The only relevant parameter at this point is the location of the file, called, in GATE terminology, the SourceUrl. By clicking on the yellow folder symbol that appears at the right side of the first row in the 'Value' column, an 'Open file' dialogue window will open.

By selecting a file and clicking on 'OK' it will be loaded into GATE, as a Language Resource. By double clicking on the document in the left pane, it will be loaded into GATE's main pane, in an Annotation Editor. To view the annotations, click on 'Annotations' and 'Annotation Sets' in the Annotation Editor's top bar. A bottom- and sideframe will open, in which the annotations will be displayed.

There may be an Annotation Set called 'Original markup annotations'. This set usually contains the html or xml tags that were found in the file.

2.3.5 Manual annotation

Load the file that is to be annotated as a Language Resource. By double clicking on the document in the left pane, it will be loaded into GATE's main pane, in an Annotation Editor. Supposing there is an Annotation Scheme loaded as well, this is how manual annotation is done. - Select the text you want to annotate - Click on the appropriate concept in the Annotation Scheme on the right

2.3.6 Automatic annotation

Run the corpus pipeline (application). Double click on the loaded application in the left pane of GATE's window. Then choose 'run'. The corpus that was loaded will be annotated automatically.

2.3.7 Viewing annotations

To view the annotations, click on 'Annotations' and 'Annotation Sets' in the Annotation Editor's top bar. A bottom- and sideframe will open, in which the annotations will be displayed. There may be an Annotation Set called 'Original markup annotations'. This set by default contains the html or xml tags that were found in the file, and is generated automatically by GATE. If the box in front of an annotation concept is checked in the 'Annotation Sets' frame (in the right part of GATE's main window), the annotations of that type are highlighted. The 'Annotations' (in the bottom frame) show the Type, Set, Start position, End position and feature structure of the annotations.

NOTE that, as a 'feature' of GATE, annotation types that do not occur in the text (i.e. there are no instances of that type in the text), are not displayed in the 'Annotations Sets' frame.

2.3.8 Saving data in datastores

Language Resources can be saved together in a Data Store. This can be useful if you are working with large datasets. It is also the safest way to ensure that when you reload the files, they look exactly the same as before.

Strangely enough, you need te create an empty folder first on your computer (outside GATE, i.e. using your operating system's functionality for that), before you can create a Data Store. Once you have got this empty folder, you right click on 'Data Stores', select 'Create datastore' and choose the fresh empty folder.

To save a file/corpus to a Data Store choose 'Save to' from the menu you get by right clicking on a resource. Then choose the Data Store the resource has to be saved to (this means you need to have a Data Store open already, though it could be empty).

2.3.9 Save data as XML

Choose 'save as XML' from the menu you get by rightclicking on a corpus or document. You can do this with data that is annotated or not, though the latter doesn't make much sense.

2.3.10 Restore application from file

In order to load a corpus and an application that you already have saved, choose (under the menu 'File') the option 'Restore application from file'. You will then be asked to choose a file. The convention is for applications to have the extension .gapp (Gate Application), but this is not necessary and any extension is fine.

3 Working with Ontologies

This section deals especially with ontologies: how to create/edit them, make Annotation Schemas out of them, and annotate texts automatically with respect to these schemas.

By default, the GATE application comes without an Ontology Editor, or OntoGazetteer, because these are under separate licence. You can obtain these by contacting the GATE team <gate@dcs.shef.ac.uk> directly.

Some additional data is needed for them, such as a creole.xml file, which tells GATE what to load, and where to find it.

3.1 Load creole.xml

In the menu 'File' choose 'Load a CREOLE repository'. You then need to specify a folder in which the relevant creole.xml file resides.

NOTE: for some reason you can not see any (creole.xml) files while browsing through the directories.

NOTE also that you should specify the folder, not the file.

3.2 Ontology Editor

The ontology editor is found in the ’Tools’ menu. If you double click on the ontology in the left window in GATE, it will be loaded in the main window. This main window looks more or less the same as the Ontology Editor itself. However, there is a difference.

NOTE that, if you edit the ontology in GATE’s main window, the changes are not saved. This only works in the Ontology Editor itself.

3.3 OntoGazetteer

The OntoGazetteer is a Processing Resource in which lists of instances of ontology concepts can be loaded. For the OntoGazetteer to function properly, it needs one or more .lst files, and two .def files, usually called mappings.def, and lists.def.

3.3.1 .lst file

A .lst file is simply a file with an instance on every line. For instance persons.lst will have, on every line, the name of a person.

3.3.2 mappings.def

The mappings.def file describes the relations between the .lst files and the ontology concepts. The format is .lst file:ontology file:ontology concept.

3.3.3 lists.def

The lists.def file states the relations between the .lst files and the annotation feature the OntoGazetteer should generate. The format is: .lst file:feature

The OntoGazetteer, when run, generates annotations for every instance mentioned in the .lst files. All these instances will be annotated with the same annotation type, called ’Lookup’ (in the ’default’ Annotation Set). Every Lookup annotation will have a feature, which is its majorType, which will differ per .lst file.

This is not very useful, because every different concept from the ontology will have the same annotation type (namely ’Lookup’). However, since the the ’majorType’ feature differs, we can process them further. And to do that, we need a Jape Transducer.

3.4 Jape Transducer

A Jape transducer is a Processing Resource for manipulating annotations. For instance, the annotations generated by the OntoGazetteer can be transcribed to distinct annotations. For this, the Jape Transducer needs a Jape grammar, usually stored in a .jape file. A Jape grammar describes which annotations should be changed, and how. See Chapter 6 of the GATE User Manual for further reference on Jape rules.

3.5 Creating a pipeline

Once a Processing Resource is loaded, it can be included in a pipeline. To do so, right click on 'Applications' in the left pane, and choose one of the approriate options. A particularly useful one is the 'Corpus Pipeline' that lets you run an application on an entire corpus. A pipeline is created by dragging the Processing Resources into it. They will be run one after another.

NOTE that in order to run the OntoGazetteer, only the OntoGazetteer needs to be selected, and not the Hash Gazetteer.

3.6 Portability

There are ways in GATE that make it possible to refer to files other than by their absolute filenames. You may want to use this, e.g. because you want some resources to be portable, i.e. usable for people on other computers, with possibly other file structures.

3.6.1 Language resources: gate:/ path names

In the bin/ directory of GATE you will find a gate.jar file. This is actually a zipped directory, which, for some reason, is called .jar. In this directory is a directory gate/ (again), in which a directory resources/ resides. This is the directory referred to as gate:/ in pathnames of resources. If you want to refer to some file with a geta:/ pathname, you should unzip the directory gate.jar (as e.g. gate/), go to the directory gate/resources/ (so, by way of example, the full pathname after unzipping might be:

```
C:\Program Files\GATE
2.2\bin\gate\gate\resources
```

copy a file there called abc.html. After you have zipped the entire thing again, you have made sure the zipped file is called gate.jar, and you have put it where it was originally stored (i.e. the bin/ directory), you can refer to the file as gate:/abc.html. NOTE that this only works for Language Resources.

3.7 Processing resources: Saving application state

See the GATE User Manual, section 2.18 ”Save Resource Parameter State to File”.

3.8 How to generate annotations automatically

Now that all the building blocks have been discussed, let us describe how to actually construct something useful out of it.

Our aim is to have an Annotation Set that contains the concepts out of the ontology. First, we make an OntoGazetteer. This we build from our ontology, and lists.def and mappings.def file. These files can be situated anywhere. If they are in the gate.jar file, we refer to them with a gate:/ pathname. Otherwise, we can pick them in the 'browse file' window, or type in a 'traditional' pathname. Suppose we run the OntoGazetteer on our corpus. This would produce, as described above, 'Lookup' annotations like this:

```
Type
Set
Start
End
Features
Lookup
Default
281
302
{majorType=Department}
```

Now we want to have a Jape rule that turns this annotation into an annotation of type 'Department'. So it should select every annotation with 'Department' as its majorType, and convert it. This is the rule that does it:

```
Rule: departmentsRule
(
{Lookup.majorType == Department}
):departmentslabel
-->
      :departmentslabel.departments = {rule = "departmentsRule"}
```

For the exact syntax for Jape rules, we refer to the manual again, but it will be clear more or less what happens here.

This then is all we need. We build a corpus pipeline with first the OntoGazetteer, and then the Jape Transducer, and if it is run, the corpus will get annotated automatically.