

INFORMATION EXTRACTION FOR ELEGISLATION

A Thesis
Presented to
the Faculty of the College of Computer Studies
De La Salle University Manila

In Partial Fulfillment
of the Requirements for the Degree of
Bachelor of Science in Computer Science

by

LIM, Brian Kent
MIRANDA, Angelo Crisanto
TROGO, Janine
YAP, Fe Eleanor

Allan BORRA
Adviser

August 31, 2010

Abstract

Information extraction (IE) is the process of transforming unstructured information of documents into a structured database of structured information. This technology allowed more narrowed-down search results of documents stored in Document Management System (DMS). An IE system was developed to augment a Blue Ribbon Committee (BRC) DMS for the eParticipation Project. IE architectures were studied and related tools were identified to develop the IE system specifically for the BRC. The IE System is composed of 7 minor modules namely Sentence Splitter, Tokenizer, Cross Reference, Part of Speech Tagger, Unknown Word, Named Entity Recognition and Preparser, 3 major modules which are Semantic Tagger, CoReference Resolution and Template Filler, and 2 external modules which are Search and Evaluation modules. With the help and constant communication with the Blue Ribbon Committee, the research was able to gather documents that helped in creating the system. Also, the output is already created and extracted based on the preference of the client and that the output system is already meeting the standards requested by the Blue Ribbon Committee. Overall, the system showed favorable results in the actual testing phase which had an output of 95.42%, but when the initial format of the documents were followed, the result of the system would be 100% accurate. Upon presenting the system to the main stakeholders, they remarked that what they had seen was already beyond their expectations and they were very pleased about the outcome. There are still parts of the system which could be improved on, such as train the values of the POS Tagger and the Named Entity Recognition from the documents being fed, update the library used to open word document files, add documents and templates to the system's process, add image recognition to the system, update web crawler for more sources and improve the search ranking algorithm.

Keywords: Information Extraction, eParticipation, eLegislation, Blue Ribbon Committee, Document Management System

Table of Contents

1	Research Description	1-1
1.1	Overview of the Current State of Technology	1-1
1.2	Research Objectives	1-2
1.2.1	General Objective	1-2
1.2.2	Specific Objectives	1-2
1.3	Scope and Limitations of the Research	1-2
1.4	Significance of the Research	1-4
1.5	Research Methodology	1-4
1.5.1	Research Analysis	1-4
1.5.2	System Design	1-5
1.5.3	System Implementation	1-5
1.5.4	System Testing	1-5
1.5.5	Documentation	1-6
1.5.6	Integration	1-6
1.5.7	Deployment	1-6
1.5.8	Calendar of Activities	1-6
2	Review of Related Literature	2-1
2.1	Information Extraction	2-1
2.1.1	Domain Independent Information Extraction Systems	2-2
2.1.2	Domain Dependent Information Extraction Systems	2-4
2.1.3	Information Extraction Architectures	2-6
2.1.4	Information Extraction Tools	2-9
2.2	Evaluation of Systems	2-12
2.2.1	ANNIE	2-13
2.2.2	Legal TrUTHS	2-13
2.2.3	Simple Information Extraction System	2-13

2.2.4	Summary of Evaluation of the Systems	2-14
2.3	Summary of Existing Systems	2-14
3	Theoretical Framework	3-1
3.1	Tokeniser	3-1
3.1.1	Tokeniser Rules	3-1
3.1.2	Token Types	3-1
3.1.3	English Tokeniser	3-2
3.2	Sentence Splitter	3-2
3.3	POS Tagger	3-3
3.3.1	ANNIE's POS Tagger	3-3
3.3.2	Cross Reference	3-4
3.3.3	LingPipe's POS Tagger	3-4
3.3.4	LingPipe's Noun and Verb Chunking	3-5
3.4	Semantic Tagging	3-5
3.5	Named Entity Recognizer	3-6
3.6	Coreference Resolution	3-6
3.6.1	Ruslan Mitkov	3-7
3.6.2	Lappin Leass'	3-7
3.7	Hidden Markov Model	3-7
3.8	Tribayes	3-8
3.9	Modified Three Way Match Method	3-8
3.10	MySQL	3-10
3.11	MyDMS	3-10
3.12	Evaluation Metrics	3-10
3.12.1	Precision	3-11
3.12.2	Recall	3-11
3.12.3	F-measure	3-11
3.13	Resources	3-11

3.13.1 Training Corpus	3-12
3.13.2 Lexicon	3-12
3.13.3 Models	3-12
4 System Objectives	4-1
4.1 System Overview	4-1
4.2 General Objective	4-1
4.3 Specific Objectives	4-1
4.4 Scope and Limitations	4-2
4.5 System Architecture	4-4
4.5.1 Pre-processor	4-4
4.5.2 Filter	4-7
4.5.3 Preparser	4-7
4.5.4 Semantic Tagger	4-7
4.5.5 Coreference Resolution	4-7
4.5.6 Template Filler	4-8
4.5.7 Evaluation	4-8
4.5.8 Search	4-8
4.6 Hardware and Software Specification	4-9
4.6.1 Development	4-9
4.6.2 Deployment	4-9
5 Design and Implementation Issues	5-1
5.1 Introduction	5-1
5.2 Data Gathering	5-1
5.2.1 Documents	5-1
5.2.2 Templates	5-2
5.3 Modules	5-3
5.3.1 Preprocessor	5-3

5.3.2	Preparser	5-11
5.3.3	Semantic Tagger	5-11
5.3.4	CoReference Resolution	5-16
5.3.5	Template Filler	5-20
5.4	Evaluation System	5-21
5.5	Entity Relationship Diagram	5-22
5.6	Class Diagram	5-23
6	Results and Observations	6-1
6.1	Back-End Evaluation	6-1
6.1.1	Testing Setup	6-1
6.1.2	System Performance Analysis	6-1
6.1.3	Modular Testing	6-4
6.1.4	Actual Data Results	6-18
6.1.5	Comparison with Legal TrUTHS Architecture	6-19
6.2	Front-End Evaluation	6-19
7	Conclusion and Recommendation	7-1
7.1	Conclusion	7-1
7.2	Recommendation	7-2
	References	Ref-1
A	Part-of-Speech Tags used in the Hepple Tagger	A-1
B	Sample Documents from the Blue Ribbon Committee	B-1
C	Sample Templates	C-1
D	Resource Persons	D-1
E	Personal Vitae	E-1

List of Figures

1.1	Research Phases	1-5
2.1	ANNIE Architecture	2-3
2.2	Legal TrUTHS Architecture	2-5
2.3	Hobb's Architecture	2-6
2.4	FASTUS Architecture	2-7
2.5	Simple Information Extraction Architecture	2-8
3.1	Hidden Markov Model Components	3-8
3.2	Modified Three Way Match and the order of Matching	3-9
4.1	System Architecture	4-5
5.1	ER Diagram	5-22
5.2	Class Diagram	5-23
5.3	Class Diagram: Sentence Splitter	5-23
5.4	Class Diagram: Tokenizer 1	5-24
5.5	Class Diagram: Tokenizer 2	5-24
5.6	Class Diagram: Cross Reference	5-25
5.7	Class Diagram: POS Tagger	5-25
5.8	Class Diagram: Unknown Word	5-25
5.9	Class Diagram: Named Entity Recognition	5-26
5.10	Class Diagram: Preparser	5-26
5.11	Class Diagram: Semantic Tagger	5-27
5.12	Class Diagram: CoReference Resolution	5-27
5.13	Class Diagram: Template Filler	5-28
6.1	Analysis of the Website	6-20
B.1	Hearing Highlights Document Page 1	B-2
B.2	Hearing Highlights Document Page 2	B-3
B.3	Hearing Highlights Document Page 3	B-4

B.4	Hearing Highlights Document Page 4	B-5
B.5	Hearing Highlights Document Page 5	B-6
B.6	Hearing Highlights Document Page 6	B-7
B.7	Hearing Invitation Document	B-8
B.8	Transcript of Hearings Document	B-9
B.9	Senate Memorandum Document	B-10
B.10	Document Evidence	B-11
B.11	Other Administrative Documents	B-12

List of Tables

1.1	Timetable of Activities (May 2009 - August 2009)	1-6
1.2	Timetable of Activities (September 2009 - December 2010)	1-7
1.3	Timetable of Activities (January 2010 - April 2010)	1-7
1.4	Timetable of Activities (May 2010 - Aug 2010)	1-7
2.1	Summary of Related Systems	2-14
5.1	Sentence Splitter 1	5-3
5.2	Sentence Splitter 2	5-3
5.3	Tokenizer 1	5-4
5.4	Tokenizer 2	5-5
5.5	Statements to find the boundaries of short form	5-6
5.6	Statements to find the list of short forms and long forms.	5-6
5.7	Conditions for getting the Formal and Informal names of the person	5-7
5.8	Conditions and corresponding actions in gathering of positions and departments	5-7
5.9	Conditions and corresponding actions in gathering the names, positions and departments in the Hearing highlights document	5-8
5.10	The heuristics for determining if the string is a department, position or a common name in the Hearing Highlihgs document	5-8
5.11	CoReference Improvement	5-8
5.12	Unknown Word Pseudocode	5-10
5.13	Agenda and Scenario Pseudocode	5-12
5.14	Committee Report Pseudocode	5-12
5.15	Hearing Highlight Pseudocode	5-13
5.16	Hearing Invitation Pseudocode	5-13
5.17	Notice of Hearing Pseudocode	5-14
5.18	Order Pseudocode	5-14
5.19	Subpoena Pseudocode	5-15
5.20	Ruslan Mitkov Rule 1	5-18

5.21 Ruslan Mitkov Rule 2	5-18
5.22 Ruslan Mitkov Rule 3	5-18
5.23 Ruslan Mitkov Rule 4	5-18
5.24 Ruslan Mitkov Rule 6	5-18
5.25 Ruslan Mitkov Rule 7	5-19
5.26 Ruslan Mitkov Rule 8	5-19
5.27 Ruslan Mitkov Rule 9	5-20
5.28 Template Filler: Normalize Name	5-21
5.29 Template Filler: Normalize Date	5-21
5.30 Template Filler: Normalize Time	5-21
6.1 Training Data Results	6-1
6.2 Actual Data Results	6-2
6.3 Sentence Splitter Modular Test	6-4
6.4 Tokenizer Modular Test	6-5
6.5 Cross Reference Person Modular Test	6-7
6.6 Cross Reference Organization Modular Test	6-8
6.7 POS Tagger Modular Test	6-8
6.8 Unknown Word Modular Test	6-9
6.9 Named Entity Recognition Modular Test	6-9
6.10 Preparser Modular Test	6-10
6.11 Semantic Tagger Modular Test	6-10
6.12 Test script for unit testing of Coreference Module with Ruslan Mitkovs rules . . .	6-13
6.13 Test script for module test of CoReference module	6-16
6.14 Test script for unit testing of Coreference Module with Ruslan Mitkovs rules . . .	6-18
C.1 Hearing Highlight sample template fields	C-1
C.2 Hearing Invitation sample template fields	C-1
C.3 Senate Memorandum sample template fields	C-2
C.4 Document Evidence sample template fields	C-2

C.5 Referral sample template fields	C-3
C.6 Notice of Hearing sample template fields	C-3
C.7 Committee Report sample template fields	C-4

1 Research Description

In this chapter, the discussions will be focused mainly on the current technologies of Information Extraction Systems, objectives, scope and limitations, and the significance of the proposed project.

1.1 Overview of the Current State of Technology

Information Extraction (IE) is the process of transforming unstructured information of documents into a structured database of structured information. An IE system consists of the following modules, according to Hobbs: text zoning, preprocessing, filtering, pre-parsing, parsing, fragment combination, semantic interpretation, lexical disambiguation, co-reference resolution, and template generation (Hobbs et al., 1992).

Basing on the general information focus of the IE system, it could be classified into two types, domain independent and domain dependent. Domain independent IE systems can handle heterogeneous texts and subject domains. IE systems of this type often use generic classification schemes which could be refined by modifying the information processing task making it demand more specific identification of semantic information (Moens, 2006). Systems like Avatar IE (Jayram et al., 2006), Generalized Hidden Markov Model (Zhong & Zhou, 2006), KNOWITALL (Etzioni et al., 2004), and ANNIE (Cheng et al., 2008) are examples of domain independent IE systems. This would also mean that they allow the acceptance of wide array of document types and could only offer shallow analysis of the documents they were tasked to handle; this is one advantage of Domain independent IE systems.

On the other hand, the other type of IE system which is domain dependent, functions as a specialized and well depicted knowledge domain therefore using a very specific set of classification rules. An example would be the system SALEM (Bartolini, et al., 2007), which is similar to Legal TrUTHS (Cheng et al., 2008), only, it specializes on documents written in the Italian language used by their government. Legal TrUTHS is rather similar to what the research is aiming to create. The system Legal TrUTHS does IE on legal documents on criminal cases in the Philippines. It takes in legal documents as input then automatically generates a template containing fields. IE is only the back-end process; what the user interacts with is actually the search module of the system, a part of the front-end process. The results of the search would be the output of the backend process which are the templates presented in the table manner (Cheng et al., 2008). Since Legal TrUTHS use a pre-defined template, it can only process criminal law documents.

In the Philippines, not all sects of the government are equipped with modern ways of documentation. One example would be the Blue Ribbon Committee(BRC) (*Commission on Information and Communications Technology - Programs and Projects*, 2009). Currently, the Blue Ribbon Committee keeps categorized hard copy of documents, but there are some documents that are in soft copy but are in various formats, meaning, some are in word document and some are in wordperfect. These documents are public documents which means that people from different sectors(BRC, NGOs, citizenry) could retrieve and access these documents. In line with this, the Blue Ribbon Committee does not have any system helping them retrieve documents quickly and accurately. Systems doing such are called document management systems (DMS), a technique for computerized storing and retrieving of documents in different kinds of

formats. An example of DMS is RFID Document Management System for inventory management. However, it is important to note that even if there is a DMS that is helping the Blue Ribbon Committee, people from NGOs and the citizenry could not be able to access the documents they want to view quickly.

The problem is how the documents would be accessed faster in the Document Management System through more narrowed-down search results. This is specifically for documents in a Document Management System for the Blue Ribbon Committee of the Philippine Senate.

1.2 Research Objectives

In this chapter, the general objective and specific objectives will be discussed.

1.2.1 General Objective

The research aims to develop an IE system for documents maintained by the Blue Ribbon Committee.

1.2.2 Specific Objectives

1. To study IE architecture and techniques that are present today;
2. To assess various tools and resources that will be of help in the IE process;
3. To differentiate the characteristics of automated validation systems and algorithms of existing IE systems;
4. To identify and classify accordingly the contents of the legal documents maintained by the Blue Ribbon Committee;
5. To describe the components of the Back-End process of the system, Data Management System and Database;
6. To determine the fields that will be used by the IE Engine to categorize documents and extract relevant information;
7. To compare the performance of systems given specific algorithms using benchmark calculation; and
8. To name the components of the Front-End process of the system that will be able to search and alert the relevant parties or stakeholders.

1.3 Scope and Limitations of the Research

The research's aim was to search for algorithms, techniques, tools and resources of IE on documents which were applied on the proposed system. Some tools that were perceived of as useful

to the project were machine learning algorithms that could be able to generate extraction rules which could be useful for POS tagging and named entity recognition. Some machine learning algorithms that might be useful to the research would be; Transformation-Based Learning, Maximum-Entropy Model, Support Vector Machine, and Hidden Markov Model. A problem the research perceived in developing the system would be finding out how to know what a pronoun refers to and the solution to this is the Anaphora Resolution (Sayed, 2003). Some algorithms that could address to the Anaphora Resolution would be; Ruslan Mitkov's Algorithm and the Lappin and Leass' Algorithm (Cheng, Cua, Tan, & Yao, 2008).

The research included studying IE architectures that are available today. The studied the general architecture of IE made by Hobbs which is composed of ten modules namely text zoner, preprocessor, filter, pre-parser, parser, fragment combiner, semantic interpreter, lexical disambiguation, co-reference resolution and template generator (Hobbs, 1993). The text zoner is the module that turns text into segments by separating the formatted from the unformatted part. Preprocessor is the module that turns the output of the text zoner, text segments, into a sequence of sentences in which these sentences are a sequence of lexical items, wherein these lexical items contains words with its lexical attributes. Filter is the module that removes the irrelevant part of the text. Pre-parser is the module that classifies the lexical item's small scale structure. Parser is the module that produces the parse tree for the sentence from the sequence of lexical items. Fragment Combiner turns the set of parse trees produced by the parser to a parse tree for the whole sentence. Semantic Interpreter is the module that turns the parse tree into a semantic structure. Lexical Disambiguation removes the ambiguity of the structure produced. Co-reference resolution identifies different explanation of the same unit in dissimilar parts of the text which turns the tree-like structure into a network-like structure. Lastly, the Template Generator produces the template from the semantic structures.

The study identified and classified the contents of the legal documents maintained by the Blue Ribbon Committee. The research achieved by dialogues with key informants of the Blue Ribbon Committee, key processes and documents, identified documents by the Focus Group Discussion. The documents to be included in this study are the following: hearing highlights, hearing invitations, senate memorandums, documented evidences, requested administrative documents, complaints, endorsements, referrals, notification or notice of hearings and committee reports.

The research studied the Back-End process namely DMS and Database. Although it is assumed that the DMS is already in place, and the e-documents are already available for processing, but the research would be participating in the design which implements the Case Management System needed by the Blue Ribbon Committee.

The research also tackles the components of an IE system, which include six components, tokenization, lexical and morphological processing, parsing, co-reference, domain specific analysis and merging partial results. The tokenization component is the one that splits the input text into sentences and tokens. In lexical and morphological processing the tokens that were determined by the tokenization component is then checked in a dictionary in search of their possible part of speech and other features for the subsequent processes. The third component which is parsing identifies the structure of the document. Co-reference on the other hand, handles the reference of a noun phrase if it refers to another same entity noun phrase. Domain Specific is the fourth component and it adds the other semantic and syntactic features of the text. For the last component, merging partial results, it uses an algorithm to combine templates that are connected with each other.

The research studied how other IE systems assessed their output and validation systems in which these systems use. Other than focusing on the IE algorithms that are present today, the research also focused on systems that extracts from highly structured documents.

The research also defined the components of both search engines and alert systems that would alert involved parties and stakeholders. For the search engine, there are four components, namely WebCrawler, database, search algorithm and ranking algorithm. The first component, WebCrawler, is the one that gathers the data for the search engine. Next in line is the database, it is where the retrieved information from the WebCrawler is stored. As for the search algorithm, it interprets the words that the user types in the search box. For the last component, ranking algorithm, it is the component that ranks the results of the search on which is the most important.

The research also studied different evaluation systems and evaluation metrics. The evaluation systems that the research would study AnnotationDiff Tool and Benchmarking Tool. The evaluation metrics that have been studied were Precision, Error rate, Recall, and F-measures.

1.4 Significance of the Research

Currently, the Blue Ribbon Committee store documents as hard copies thus, reviewing a certain case would result to finding and reading loads of documents. This method consumes a lot of time since one has to search every part of a document and worst case, lead to an irrelevant output. The system will relieve them of the tedious process and replace it with an automated one. The system is capable of extracting relevant information and displaying results in an organized manner therefore, saving time. This research would also contribute to the study of Natural Language Processing through finding a way how to process combination of languages (i.e. a document that uses both Filipino and English).

1.5 Research Methodology

This section enumerates the activities that the research performed in order to accomplish the proposed thesis. The group has chosen to use the Spiral Model because the research saw it fit for the project since the overall scope of e-participation is quite large. The research preferred that a project this large would undergo continuous validation, verification and feedback gathering in order to continually modify specific parts so the requirements and goals of the research would be met fully.

1.5.1 Research Analysis

The research in this phase studied the different algorithms that can be used in order to implement an information extraction system. These algorithms include machine learning algorithms and algorithms that would be able to provide Anaphora resolution. The research also studied the format and fields of the documents that the proposed IE system will process. This phase also includes regular consultation with resource persons, especially the thesis adviser. This was done in order to update the thesis adviser on the progress of the research.

1.5.2 System Design

Using the findings from the research analysis phase, the research chose the appropriate algorithms and architecture they needed to use in order to come up with a feasible research output. The research modified the chosen architecture to better suit their system.

1.5.3 System Implementation

Using the design and the chosen software development lifecycle, the research has done actual coding of the proposed solution stated above. Debugging was also implemented in this phase to make the system more reliable and effective.

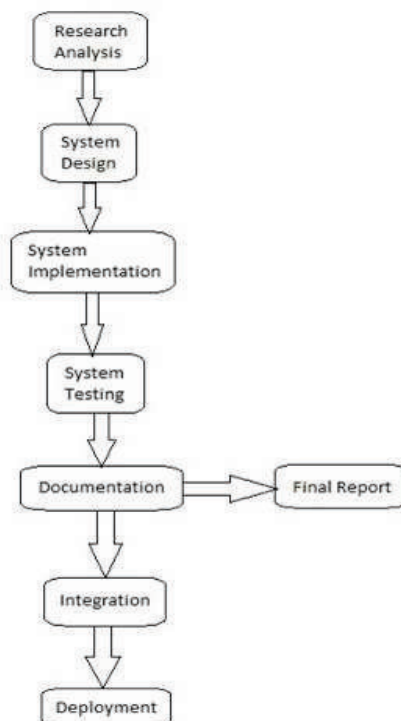


Figure 1.1: Research Phases

1.5.4 System Testing

To make sure that the quality of the system is effective, testing has been done in this phase. This phase was done to make sure that the research will use the correct algorithms in implementing extraction process and to check if the system is ready to be deployed and used with minimal or no errors. Black Box, Modular, Unit and Integration Testing would be used accordingly to check the accuracy and quality of the system.

1.5.5 Documentation

Documentation has been done all throughout the implementation and testing process to keep track of the modifications done in the system.

1.5.5.1 Final Report

The final report has been done upon finishing the project, which means that all of the stages are successfully completed and accurate. It contains the final functions and reports of the systems.

1.5.6 Integration

Our system would be a part of a larger system, so in this part the groups had integrated all the parts together. It would be the integrated system that will be deployed to the Blue Ribbon Committee.

1.5.7 Deployment

In this stage, the system would be deployed to a client specifically the Blue Ribbon Committee. The functions of the system would also be taught to the users in this stage to make them comfortable in using the system.

1.5.8 Calendar of Activities

Tables 1.1, 1.2, 1.3 and 1.4 are showing Gantt charts of the activities. Each bullet represents approximately one week worth of activity.

Table 1.1: Timetable of Activities (May 2009 - August 2009)

	May 2009	June 2009	July 2009	Aug 2009
Research and Analysis	•	•••	•••	•••
System Design				
System Implementation				
System Testing				
Documentation	•	•••	•••	•••
–Final Report				
Integration				
Deployment				

Table 1.2: Timetable of Activities (September 2009 - December 2010)

	Sept 2009	Oct 2009	Nov 2009	Dec 2009
Research and Analysis	•••	••		
System Design	••••	••••	••	••
System Implementation			•	•
System Testing				
Documentation	•	•	•	••
–Final Report				
Integration				
Deployment				

Table 1.3: Timetable of Activities (January 2010 - April 2010)

	Jan 2010	Feb 2010	Mar 2010	Apr 2010
Research and Analysis				
System Design				
System Implementation	••••	••••	••••	••
System Testing	•	•	••	••••
Documentation	•••	•••	•••	•••
–Final Report				
Integration		•	••	••••
Deployment				

Table 1.4: Timetable of Activities (May 2010 - Aug 2010)

	May 2010	June 2010	July 2010	Aug 2010
Research and Analysis				
System Design				
System Implementation				
System Testing	••••	••••		
Documentation	•••	••••		
–Final Report		••	••••	
Integration	••••	••••	••••	
Deployment				••

2 Review of Related Literature

This chapter discusses the related literatures and systems that the research have studied.

2.1 Information Extraction

Information extraction is the process of filling the fields and records of a database from unstructured or loosely formatted text. The process is divided into five sub-processes: segmentation, classification, association, normalization and deduplication.

In the process of segmentation the start and end boundaries of the text snippets that will fill a database field is searched for. After segmentation comes the classification sub-process where the system determines which database field is the appropriate destination of each text segment. Association follows classification. In the association sub-process, the system determines which fields belong together in the same record. The information is then put into a standard format in which it can be reliably compared to other information. This is done in the normalization sub-process. Deduplication then collapses redundant information so the database will not contain any duplicate records.

IE can be implemented in different ways. The use of regular expressions is one, it is used only if the extraction task is simple. If the extraction would be from moderately more complex text sources with sufficient formatting regularity, then it can be addressed accurately with hand-tuned, programmed rules. Machine Learning methods such as decision trees, if-then-else rules tune their own rules or parameters in order to maximize its performance on a set of example texts that have been correctly labelled by hand. In other words, instead of trying to tune the complex extraction rules manually, you show the machine what to do on specific example texts by performing the extraction task yourself. The machine then generalizes from these examples, appropriately tuning its own rules and parameters (Mccallum, 2005).

Distillation is another process of extracting relevant information from massive document sources. Distillation has two approaches; one is by document retrieval in response to a query. The other is by classification to extract sentences as snippets. The set of sentences is tagged with respect to relevance and then reduced by finding and removing redundancies.

Techniques such as link analysis, query expansion and keyword search are used to reduce the size of the target quantity to the extent required by information extraction. Searches in the information extraction stage can be augmented with semantic information to be precise. This is called semantic search. The corpus being further reduced; the output is then used as input in the co-reference analysis level. This stage basically enables queries on specific entities. Semantic search results are documents, while co-reference analysis searches for facts about specific entities.

As written earlier, there are two types of Information Extraction Systems, namely, domain dependent and domain independent. Systems under these two types would be discussed below.

2.1.1 Domain Independent Information Extraction Systems

In this part, the research would be discussing two different domain independent extraction systems namely ANNIE and AVATAR.

For the first system, we would be discussing about ANNIE which uses GATE, an architecture development environment and framework for building systems that process human language has been in development at the University of Sheffield since 1995. It is distributed with an Information Extraction set called ANNIE, standing for A Nearly-New IE system. Annie is built to be a portable IE system. It was actually intended to be used in various applications, on various kinds of text and for various purposes. Given that it is portable, it is implied that the system must cope perfectly with documents in different formats and could process data in multiple languages and process large data volumes without crashing and at high speed.

ANNIE also has the power to evaluate its output using its three modes of evaluation. The first mode is comparing a manually annotated document with the document that is being processed. Another mode compares the currently processed document and the previously processed document to see its progress. The third mode of evaluation compares the previously evaluated document with a document annotated by a user. To see how correct the output of their system is, they use the following metrics in evaluation: recall, error rate, precision, f-measure, false positives and inter-annotator agreement. The inter-annotator agreement simply means that it should be annotated by more than one expert so it would be comparable, if by any chance, the expert does not agree with one another, then a system could not possibly annotate it correctly. The ANNIE system also considers three criteria of correctness which are: strict, lenient and average. Strict means that all partially correct responses are considered incorrect. On the other hand, lenient considers all partially correct responses to be correct. The average measure gives a half weight to partially correct responses.

ANNIE has been modified through the years in order to support a diverse set of languages with minimum overhead. The boon of having low-overhead portability such as that ANNIE has is that it enables quick deployment of IE tools with an acceptable performance (Maynard & Cunningham, 2003). The modules of ANNIE shown in Figure 2.1 (Cheng et al., 2008) are the following: Unicode Tokenizer, Gazetteer lookup, Sentence Splitter, Part of Speech tagger, Semantic Tagger, Name matcher, and the pronominal co-reference. To generate the template to be used the document undergoes those modules. The Unicode tokenizer which is needed for POS tagging splits the given text into simple tokens. The goal of the this process is to lessen the work of the tokenizer and give greater flexibility through placing the huge amount of work on the grammar rules which are more adaptable. The output tokens of the Unicode tokenizer are then passed to the gazetteer lookup module which acts as a list where it represents a set of names. Tokens that are matched by these machines can be annotated with features based on the provided rules. The grammar rules specify the types to be identified in particular circumstances. The annotated tokens produced by the second module will then be passed to the sentence splitter module which then segments the text into sentences. This is required for the tagger. The splitter could use the list of abbreviations from the gazetteer to help tell if the analyzed text is a sentence-marker. This module is domain and application-independent. Set of text segments from the sentence splitter will be passed on to the speech tagger module. The tokens from the sentences are given part of speech tags. The rule set for the tagger can be developed manually, however, finding existing rule sets is more recommended since the existing ones are already trained from a large corpus that consists of more than a million words. POS tagged tokens are produced and passed onto the semantic tagger module that contains

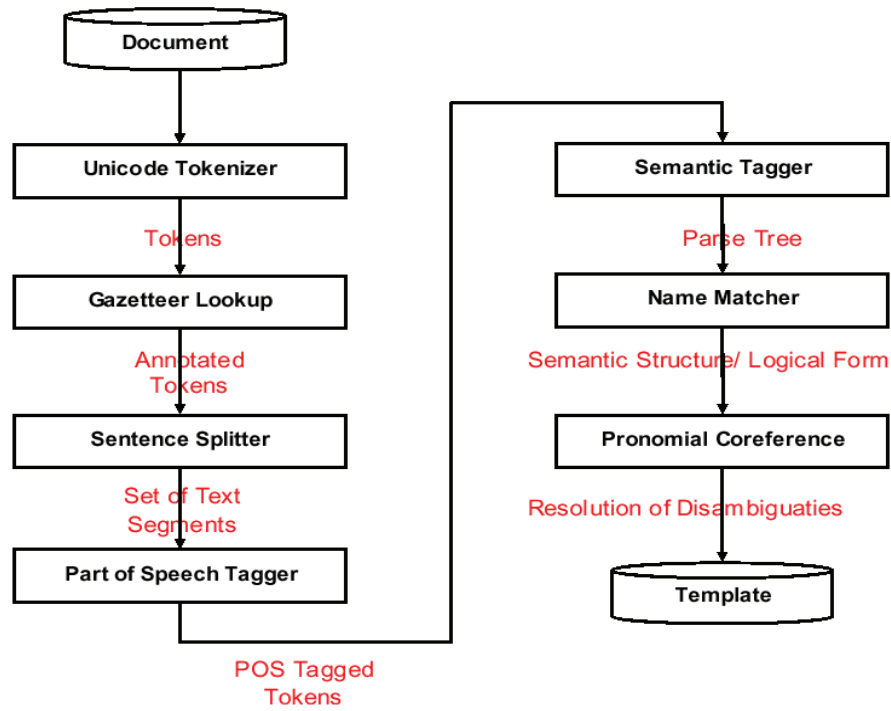


Figure 2.1: ANNIE Architecture

rules which act on annotations or classifications assigned in earlier phrases. With the previous information, the module can then produce outputs of annotated entities. The semantic tree produced by the current module is then passed to the name matcher module also known as the orthographic co-reference which adds identity relations between named entities found by the semantic tagger module. This is done in order to perform co-reference. It can assign a type to an unclassified proper name using the type of a matching name. The matching rules are only invoked if the names being compared are of the same type. With this module, re-categorization of a previously classified name is prevented. A semantic-structure or a logical form is then passed onto the last module which is the pronominal co-reference which performs anaphora resolution. The last module has three submodules: the quoted text submodule which identifies quoted fragments in the text being analyzed then the identified fragments are used by the pronominal coreference module for the proper resolution of pronouns. The pleonastic it submodule matches the redundant occurrence of the word “it”. The pronominal resolution submodule anticipates each pronoun then sees what possible candidate antecedents can be used for this kind of pronoun and chooses the best candidate if there is any.

On the other hand, we have AVATAR Information Extraction System which is created by IBM Almaden Research Center. It uses rule-based techniques to extract information from text documents. It also uses “probabilistic database techniques” to do the extraction process. These techniques involve creating frameworks that would be used to map queries in the database.

AVATAR IES could be divided into two categories of annotators, (1) Base Annotators and (2)

Derive Annotators. Base Annotators process over the text and derive annotators on the other hand process on the text. Every annotator would produce annotations which would be defined with a set of operations, or also called as rule. If there is a set of rules in an annotator it is called meta-rule.

For now, AVATAR uses UIMA workflow engine to produce the annotators. The produced annotations are then stored in a annotation storage which was created using a relational database. To prevent problems, AVATAR IES thinks that filtered inputs have equal precision.

AVATAR IES uses rule-based annotator template, in this template it could show the three kinds of rules. Namely, feature extraction rules, candidate generation rules and consolidation rules. In feature extraction rules, in which it uses rules to create a set of features. For the candidate generation rules, it uses rules to create a set of candidates and lastly, we have consolidation rules which use rules to create results.

This system also has disadvantages and one of them are, the annotations used by the derived annotator is from the base annotator, which means that if the annotations produced by the base annotators are wrong then the result would now be affected (Jayram, Krishnamurthy, Raghavan, Vaithyanathan, & Zhu, 2006).

2.1.2 Domain Dependent Information Extraction Systems

As of this part, the research would be discussing the two systems that depend on the domain of the documents it would be extracting namely SALEM or Semantic Annotation for Legal Management, and Legal TrUTHS.

The first domain dependent information extraction system, SALEM, is an NLP system used on Italian law paragraphs to produce a semantic structure through tagging. The system does this with the integration of NLP and information extraction technology. Unlike ANNIE and AVATAR that was stated above, SALEM is a domain dependent system that performs information extraction.

Part of the framework of SALEM is the twofold task it performs. First of all, SALEM assigns each law paragraph to a given legislative provision type. A legislative provision type is a part of a law section wherein it expresses permission or an obligation for someone to perform or not to perform a certain action. The next task would be to tag parts of the paragraph with domain-specific semantic roles which in effect identifies the legal entities pointed out in the legislative provision. Legal entities would refer to actors, actions and properties.

As briefly explained above, SALEM takes in single law paragraphs and outputs a semantic structure by tagging. This is achieved by following a two-stage strategy. The first step involves a parsing system which pre-processes each law paragraph to give a somehow shallow syntactic analysis. This strategy is called syntactic pre-processing. The second and the last strategy called the semantic annotation, feeds in the syntactically pre-processed text from the first strategy. This in effect makes it more explicit from the original implicit conveyance.

After the preliminary evaluation, SALEM's results were proved to be very promising. The system was evaluated on approximately 500 law paragraphs. The problem is that SALEM is limited only to a few ontological provisions. Their aim is to expand the ontology of provisions in order

to accommodate new provision types (Bartolini, et al., 2007).

The next domain dependent system which is quite similar to SALEM is called Legal TrUTHS. It is a system that uses information extraction to extract relevant information from unstructured documents and translates it into helpful structures. Like SALEM, Legal TrUTHS is a domain dependent system. However, Legal TrUTHS specializes only in criminal type of law documents. Legal TrUTHS features a back-end and a front-end module. The front-end provides the search-engine for the user to be able to find the document that is mostly related in his query while the back-end module is where the information extraction takes place. The system accepts criminal

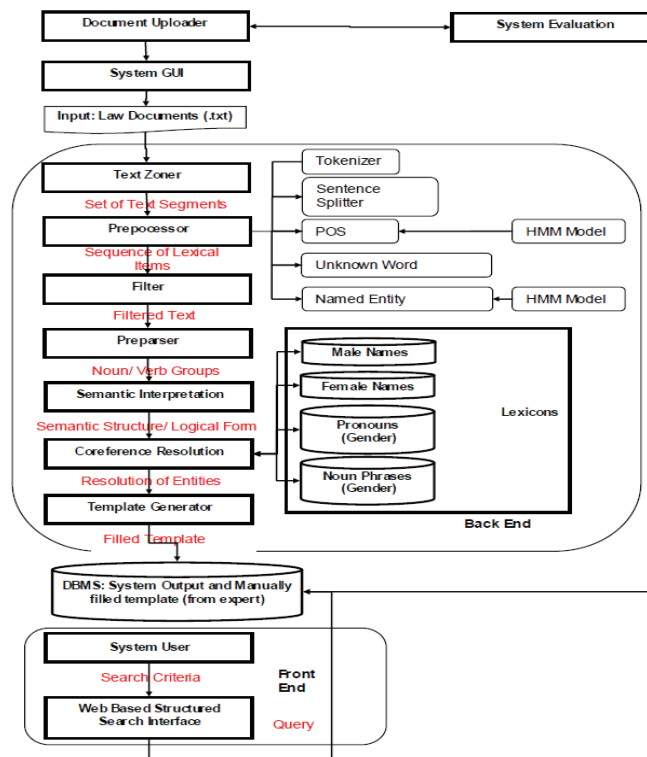


Figure 2.2: Legal TrUTHS Architecture

law documents and processes it through seven modules which could be seen in Figure 2.2 (Cheng et al., 2008). First, the document is processed through the text zoner. It identifies then classifies the different parts of the document into different regions. This is necessary as to reduce the load on the next modules. The output of the text zoner would then be taken by the preprocessor module as input. The preprocessor has three sub-modules: tokenizer, sentence splitter, POS tagger, unknown word heuristics, and named entity recognition. The purpose of the preprocessor is mainly to output lexical items; tokens with lexical attributes. The next module would be the filter module. The filter modules job is to prune out the unnecessary sentences which the system would unlikely process in later modules. The parser would then take in the text segments with their respective lexical attributes and try to group the words into phrases such as noun phrases and verb phrases. This module is necessary because it will make the job of the semantic interpreter and conference resolution easier. After the parser, the next would be the semantic interpreter. The semantic interpreter would process the words or phrases or

tokens to remove from them the ambiguities. Also, the semantic interpreter would decide which tokens are most likely to be candidates for the fields in the template which would be based on rules and pattern matching heuristics. The coreference resolution will then resolve coreference for basic entities. And the last module would be the template generation module. The template generation module is where the template is filled up and the extracted information would be normalized to follow one format. The output template would then be stored in the database.

Legal TrUTHS also provides the evaluation module. This module is tasked to check the correctness of the systems output. The module compares the result to the one created by an expert (Cheng et al., 2008).

2.1.3 Information Extraction Architectures

In this section the research would be discussing different architectures used in Information Extraction. These architectures are Hobbs' Architecture, FASTUS Architecture, and Simple Information Extraction Architecture. According to Hobbs, shown in Figure 2.3 (Hobbs, 1993)

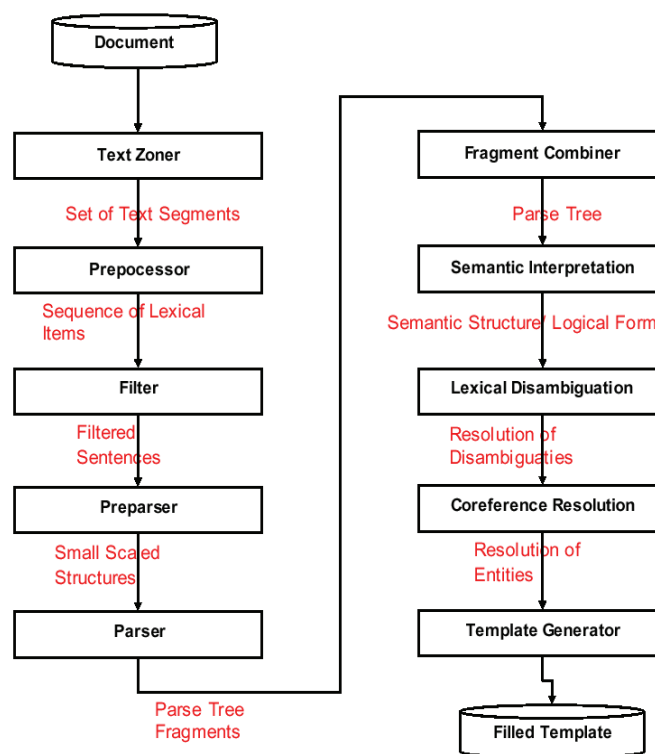


Figure 2.3: Hobb's Architecture

different systems may have an architecture that consists of different modules but he claims that even if their architecture modules differ in some ways, basically they still must consist of the following: Text zoner, pre-processor, filter, preparser, parser, fragment combiner, semantic interpreter, lexical disambiguation, coreference resolution or discourse processing and a tem-

plate generator. The given text will first undergo the text zoner which turns a text into a set of text segments to be processed by the pre-processor afterwards. The pre-processor turns a text or text segment into a sequence of sentences. Each sentence is a sequence of lexical items wherein a lexical item is a word that comes with its lexical attributes. The set of sentences will then be passed on to the filter turning those into a smaller set of sentences through filtering out the irrelevant sentences. The preparer takes in the sequence of lexical items and tries to identify different reliably determinable small-scale structures. The next module is the parser that accepts a sequence of lexical items or small-scale structures such as phrases and outputs a set of parse tree fragments that may or may not be complete. The fragment combiner turns a set of parse trees or logical form fragment into a parse tree or logical form for the whole sentence. The parse tree or parse tree fragments is passed on to the next module which is the semantic interpreter which generates a semantic structure or logical form to be passed to the lexical disambiguation module. The lexical disambiguation module turns a semantic structure containing general or ambiguous predicates into a semantic structure that will, after being processed by the module, contain specific unambiguous predicates. The output of the lexical disambiguation module will then be inputted to the co-reference resolution or discourse processing module which transforms a tree-like structure to a network-like structure through identifying the different descriptions of the same entity in the different parts of the text. Finally, the output of the current module is processed by the last module which is the template generator which is in charge of deriving the templates from the semantic structure (Hobbs, 1993). Another ar-

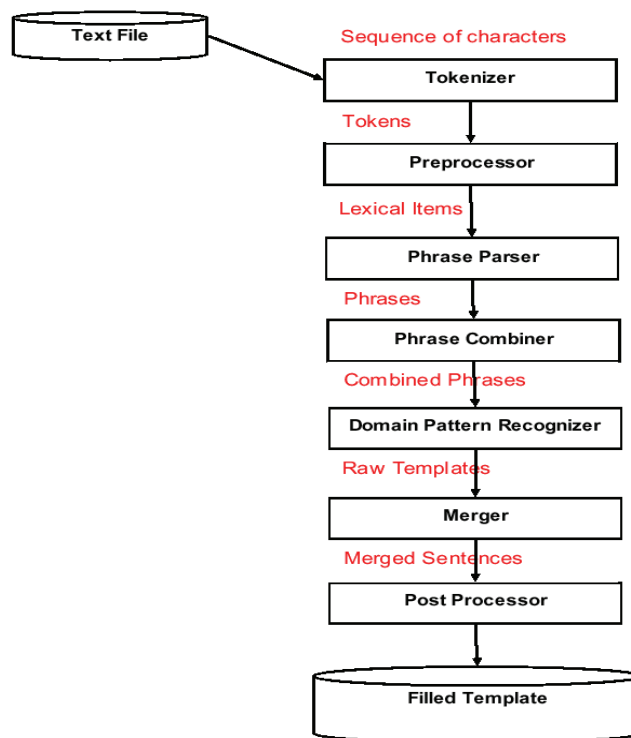


Figure 2.4: FASTUS Architecture

chitecture shown in Figure 2.4 (Appelt et al., 2004) is the FASTUS Architecture. The FASTUS Architecture is summarized into three phases namely, text scanner, finite state machines and

another finish state machine. For the first phase, text scanner, it scans the text which identifies proper names, correct spelling, and preprocessing tasks to make sure that the input text is in a standardized format. The next phase, finite state machines, it accepts a sequence of words from the input text and outputs a sequence of determiners, prenominals, head nouns, auxiliaries, adverbs, particles, prepositions, conjunctions, and genitive markers. It is then filtered to include the longest elements. For the last phase, it is passed to another finite state machine, in this transition it creates a structure which could be said to be a proto-template, when it already reaches the final state, the templates that were produced are saved and merged with other templates of the same sentence. After merging, it is done on every sentence, and then the templates are merged together to form the whole text's template.

One more architecture, shown in Figure 2.5 (Giuliano, 2006), is the Simple Information Extrac-

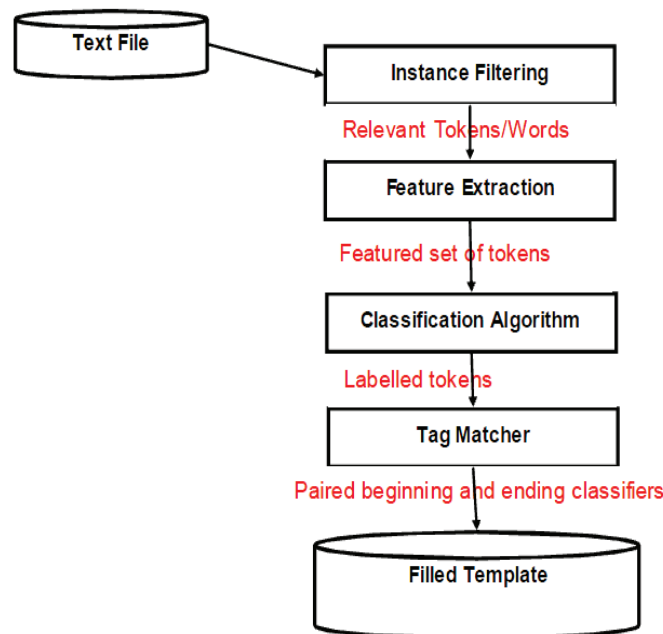


Figure 2.5: Simple Information Extraction Architecture

tion Architecture. This architecture starts from the text file. The input file goes to the instance filtering phase, which removes some tokens in the set data to speed up the process. In this phase, it discards harmful parts of the text without lowering the accuracy. Instance Filtering assumes that uninformative words do not belong to the entities to be recognized. The next phase is the Feature Extraction phase in which it extracts pre-defined features for each input token that was filtered in the first phase, instance filtering. The next phase after feature extraction is classification algorithm, in which it assigns classification labels to each token that is in the set data except for the tokens that were labeled as irrelevant. The next phase to be explained is tag matcher, which pairs up classified tokens. If overlapping entities are encountered, the one with the higher score is selected. The score mentioned in the tag matcher is the entity length

probability.

2.1.4 Information Extraction Tools

In this section the research would be discussing different tools used in Information Extraction. These tools are Anaphora Resolution, Sentence Boundary Detection, POS Tagger, and Named Entity Recognition.

The algorithm of Ruslan Mitkov uses a knowledge-poor approach in order to coreference resolution. Doing this, parsing and analysis becomes unnecessary in identifying the antecedents of the anaphors. It relies on inputs that have been processed beforehand such as the part-of-speech (POS) tagger and simple noun phrase rules. It would then operate on antecedent-tracking preferences.

It examines the current sentence and the sentences that follow the current, finding noun phrases to the left of the anaphor. The noun phrases are identified whether they agree in gender and the number with the pronominal anaphor will be grouped and marked as potential candidates. Antecedent indicators such as definiteness, givenness, indicating verbs, lexical reiteration, section heading preference, non-prepositional noun phrases, collocation pattern preference, immediate reference, referential distance and term distance as indicators, which are actually heuristics. These are the basis of what scores will be assigned to the candidate. the candidate with the highest total score is proposed to be the antecedent.

Unlike the Ruslan Mitkov's algorithm, The algorithm of Shalom Lappin and Herbert J. Leass relies on measures of prominence that is derived from syntactic structure and a dynamic model of attentional state to select the antecedent noun phrase of a pronoun from a list of candidates making it able to identify intrasentential and intersentential antecedents of pronouns in the text processed. The presented algorithm consists of an intrasentential syntactic filter for ruling out anaphoric dependence of a pronoun on a noun phrase due to nonagreement of person, number, or gender features. This is a procedure for pointing out pleonastic pronouns, which are pronouns that are semantically empty. It is an anaphor binding algorithm used in identifying the possible antecedent binder of a lexical anaphor within the same sentence. It is also a procedure used in order to assign values to several salience parameters such as grammatical role, parallelism of grammatical roles, frequency of mention, proximity and sentence recency. This procedure is applied to a noun phrase that utilizes a grammatical role hierarchy, a procedure for identifying noun phrases linked anaphorically which is treated as an equivalence class for which global salience value is computed and considered as the sum of the salience values of its elements and the basis of procedure when it comes to selecting the preferred element of the list of antecedent candidates for a pronoun.

Sentence Boundary Detection tool is essential to an information extraction system. Theoretically, this tool correctly splits the text into their respective sentences. Sentence stoppers, such as periods (.), question marks (?), and exclamation points (!), usually separate a sentence from another sentence. However, in some sentences, this condition does not apply (Nadeau, 2007).

Legal TrUTHS used an EditedSentenceModel to enable the Sentence Splitter sub-module. The EditedSentenceModel was based on a sentence model for biomedical research abstracts developed by Lingpipe. The Lingpipe version is modified to include additional Filipino suffixes of locations and persons and extensions found in law documents.

Legal TrUTHS also pointed out that determining the sentence boundary based on possible sentence stoppers is insufficient. Cases such as acronyms, places, titles, quotations, and fragments that may contain sentence stoppers but do not necessarily convey an ending of a sentence. In light of this issue, Legal TrUTHS provided three data structures that could solve these special cases. The first data structure is POSSIBLE_STOPS. It is a set of possible tokens which marks the end of a sentence. The second one is the IMPOSSIBLE_PENULTIMATES which is a set of possible tokens which may not precede a sentence-ending token. And the last data structure is the IMPOSSIBLE_SENTENCE_STARTS which is a set of tokens which may not follow a sentence-ending token (Cheng et al., 2008).

A lot of natural language processing systems use POS Taggers. A POS Tagger is a tool that performs part-of-speech tagging usually by assigning a unique or ambiguous part-of-speech tag to each token in the input. After that, the output is passed to the next processing level which most of the time, is a parser (?, ?).

Balie, a baseline information extraction system shows us a process of how to assign a part-of-speech to a token or word. First, it uses a TokenList which is a list of words or punctuations and a POS Tagger called QTag. The tokens in this case are designed to be used with a bit mask technique. The part-of-speech is determined by using a 32-bit integer. Each position of the 32bit integer corresponds to a certain part-of-speech (Nadeau, 2007).

QTag, as mentioned above, is a tool that uses probabilities to tag a part-of-speech to word instead of being rule based. So QTag works by combining two sources of information, a lexicon file which contains the words and the tags with the associated frequency or probability, and a matrix file that contains the tags in sequential order from a pre-tagged training corpus. The process starts with a window of three tokens. These tokens are looked up one by one in a dictionary which was also built from the pre-tagged training corpus. If the token is in the dictionary then the tag assigned will be based from the dictionary, otherwise QTag will guess the tags possible for the tokens by using probabilities. There are three steps in accomplishing this, first is to compute for the probability of the token to have a certain tag which is P_w . P_c , the second one, is the chance that a certain tag will occur given two tags following it. Lastly, $P_{w,c}$ is the probability of such a tag sequence happening and of course, the one with highest probability will be chosen (?, ?).

Another tool called MBT, or Memory-Based Tagger is also used in some systems. The approach now is memory-based. In this sense, a set of cases is kept inside memory. Cases consist of a word with the previous and following context, and the category for the word given that context. By choosing for each word in the sentence and its context the most similar case in memory, a new sentence is tagged. This in effect would enable us to extract the category of the word (Daelemans, 1996).

Named Entity Recognition (NER) is the process of recognizing the tokens of the text in categories like name of persons, companies, organization, locations, etc. The categories that were mentioned are defined in terms of the field of the documents like law, medicine, finance, and others.

There are many Named Entity Recognition System that are present today, we will be discussing some namely Balie's and Language Independent Named Entity Recognition.

Language Independent Named Entity Recognition Systems focuses more on four types on

named entities, namely persons, locations, organizations and miscellaneous entities. The entities that the miscellaneous entities are referring to are the entities that are not included in the first three named entities mentioned above. Training data and test data would be given to the people who will be creating an NER that includes a Machine Learning Component.

Balie's Named Entity Recognition is a system that learns to know what kind of named entities are present independently. This system also features a tokenizer, sentence boundary detector, language guesser, and lastly, part of speech taggers to meet the requirements of a NER system. Balie's NER solves two limitations of NER systems, for the first, it doesn't require a human to manually label data or maybe creating gazetteers manually. For the second limitation that was solved, it does not only limit the named entities to 3 kinds, person, location and organization.

Golding and Schabes introduce the use of Tribayes, a hybrid method combining the best of the bayes method and the trigram method in context-sensitive spelling correction. They used part-of-speech trigrams in order to encode the context reducing the number of parameters and to allow training for all confusion sets allowing addition of confusion sets afterwards without any additional training making it a system that is easy to expand. Because trigrams perform poorly if it is presented words that have the same part of speech, they considered an alternative method called Bayes. Unlike trigrams, Bayes performs well when the words being processed have the same part-of-speech. Tribayes Exploits the complementarity of both by invoking each when it can produce the better results than the other (Golding & Schabes, 1996).

Circism-Tutor caters to medical students who have little experience with keyboards and who constantly inputs novel abbreviations with the mindset that he or she has given enough information to the system. The spelling correction implemented in this system is unlike those in word processors or any other authoring systems because it uses a rapid and automated correction since the system has only a few seconds to parse the student input, update the student model, plan the appropriate response, turn it to sentences and display those sentences on the screen. The filtering system for words that are possible candidates is adaptive. It begins with a wide acceptance interval and tightens the filter as better candidates appear. The parser accepts several replacement candidates for a misspelled string from the spelling corrector then it selects the best by applying syntactic and semantic rules. The selection of the candidate is dynamic and context dependent. The system limits its correction to words that are not found in the dictionary, effectively, it also deals with word boundary problems and abbreviations. It uses three way match algorithm. The spelling corrector of Circism-tutor was tested with a lexicon consisting of 4,500 words then it was tested with 102,759 words and it appears that the system can handle a very large lexicon through using context by combining parsing and spelling correction. The limitation of this system is that it cannot detect and therefore correct incorrect words that form a valid word found in the lexicon (Elmi & Evens, 1998).

Toutanova and Moore presented a method in order to incorporate word pronunciation information in a noisy channel model to correct misspelled words. They built an explicit error model for word punctuations. They claimed to have achieved a substantial performance improvement over the previous best performing models for spelling correction by modelling pronunciation similarities between words. They built a separate model for phonetic errors using the Brill and Moore learning algorithm. They made two different error models, one of which is a letter based model which is exactly the Brill and Moore model trained in on a similar data set. The second model is a phone-sequence-to-phone-sequence error model trained on the same data set as the first model, only this time, they used the pronunciations of the correct words and estimated pronunciations of the misspellings to learn phone-sequence-to-phone-sequence edits

as well as estimate their probabilities. The list of best candidates for the misspelled word inspected generated by the two models are combined using a linear model. They also built a letter-to-phone model automatically from a dictionary. This process however does not recognize whether it is a phonetic or a typographical error (Toutanova & Moore, 2002).

Information Retrieval(IR) systems are susceptible to cope with textual misspellings. Ruch evaluates the effect of misspellings on the retrieval effectiveness and used a string-to string edit distance calculus as well as a contextual method which adds linguistically motivated features to the string distance module. The two methods are implemented on separate systems. The first system is dependent on a dictionary of well-spelled tokens and it selects the top candidate based on a string edit distance calculus. The second system uses lexical disambiguation tools in order to refine the probabilities of each candidate for selection. Ruch's IR system's spelling correction is processed by computing a string edit distance between a given token and the items of a lexical list. Ruch also presented that applying spelling correction as a batch task gives the system the opportunity to replace the misspelled token by a wrong candidate (Ruch, 2002).

Simple Algorithm for Identifying acronym Definitions for identifying definitions in biomedical text

The algorithm was originally made for definitions in biomedical text particularly MEDLINE abstracts because the rate of introduction of new acronyms in biomedical literature is becoming a bigger challenge for researchers. The authors, Schwartz and Hearst claims that this is a new simple and fast algorithm for extracting acronyms from biomedical text. It is a pattern-based algorithm. It has been based on predictable patterns seen in the MEDLINE abstracts. This algorithm has been proven to have high recall and high precision, to be specific 96% precision and 82% recall on a standard test collection as compared to existing approaches which have 95% precision and 82% recall on another larger test set. The advantage of this algorithm is that it does not require training data. This algorithm can not only handle those acronyms which follows the pattern of having one letter in the acronym corresponding to the initial letter of the definition like methyl methanesulfonate sulfate (MMS), it also handles internal letters in the definition such as Gcn5-related N-acetyltransferase (GNAT) (Schwartz & Hearst, 2003).

JAWS is an API written by Brett Spell. It provides Java applications the ability to retrieve data from the WordNet database which is either in version 2.1 or 3.0. JAWS requires the installation of the WordNet program and the configuration of the location of the WordNet database in a machine before it can be used in an application (Lyle, 2009).

Wordnet, which is available at its latest version 3.0, is a huge lexical database of English words. Its development is led by George Miller. In wordnet, nouns, verbs, adjectives and adverbs are grouped into sets of related synonyms called synsets with each set expressing one distinct concept. Synsets are linked together by conceptual-semantic and lexical relations (University, 2010).

2.2 Evaluation of Systems

In this section the research would be discussing the evaluation of ANNIE and Legal TrUTHS.

2.2.1 ANNIE

ANNIE an open source information extraction system, evaluates its own output in 3 ways, all three would be comparing files. The first way compares the system output with the manually created output. Second would be comparing the system's current output and the system's output of the previous document. Lastly, the system's previous output would be compared to the manually created output. ANNIE uses different kinds of evaluation metrics to evaluate the performance of its system such as precision, recall, f-measure and false positives. ANNIE also has 3 kinds of correctness namely strict lenient and average. Strict, is the kind of correctness that only allows precise correctness, if the output is only partially correct then it will still consider it as incorrect. Lenient, on the other hand, considers partially correct output as correct already. And lastly, average, it gives the partially correct output a half score, for example if the perfect score is 10 then it will be giving the partially correct output a score of 5 (Cunningham et al., n.d.).

2.2.2 Legal TrUTHS

Legal TrUTHS uses precision, recall, and f-measure to evaluate the performance of the system. The system uses Longest Common Subsequence to check the accuracy of the output. There were two approaches studied for Longest Common Subsequence. The straightforward approach is Nave Approach to LCS, this approach has two parameters x and y, it checks the subsequences of each and then gets the longest from checking if all subsequences of x could be found in y. But, this method could not bear with high values because of its speed. The next approach would be addressing the problem, this approach is called the dynamic programming approach, which uses recursion, it will have conditions to whether continue or stop the recursion. After getting the length of the longest subsequence, the length would be over the correct answer's length and then multiplied by 100 to get the percentage of the output (Cheng et al., 2008).

2.2.3 Simple Information Extraction System

Simple Information Extraction (SIE) is an information extraction system using a supervised machine learning approach. It extracts entities which are domain-specific from documents. SIE was also designed to be easily adapted to another task and domain. As a support to this claim, several tasks in various domains and languages were used in conducting the experiment.

SIE was experimented on the following IE benchmarks: JNLPBA, CoNLL 2002 2003, TERN 2004, and Seminar Announcements. JNLPBA is the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications. The data set has several MEDLINE abstracts from the GENIA project in which annotations consist of five entities: DNA, RNA, protein, cell-line, and cell-type. CoNLL 2002 2003 on the other hand, has Dutch and English data sets with four entities namely, persons, locations, organizations, and miscellaneous names of entities. TERN 2004 requires systems to detect and be able to normalize expression occurring in English text. Lastly, The Seminar Announcements includes in its collection several electronic bulletin board postings. Annotations for documents include entities: speaker, location, time, and etime.

As said above, the SIE was experimented using the IE benchmarks. Also the difference in filtering strategies were acquired by using CC, OR and IC filters. The CC or Correlation Coefficient uses a statistic in measuring the dependence between a term and a category in which case, terms that are less likely to express relevant information can be found. OR or Odds Ratio uses the assumption that a term is not informative when its probability of being a negative example is higher than the probability of it being a positive example. Lastly, the IC or Information content is based on the frequency of the terms following the assumption that frequent terms are not relevant.

The results of the experiment are in terms of filtering rate, recall, precision, F1, and computation time for JNLPBA, CoNLL-2002, CoNLL-2003, TERN, and SA using different filtering strategies. The comparisons will show the changes in the computation time and accuracy of the system (Giuliano, Lavelli, & Romano, 2006).

2.2.4 Summary of Evaluation of the Systems

With reviewing both systems, the common metrics used in measuring the performance of an information extraction would be precision, recall and f-measure. Precision is the number of correctly identified items over the total number of items. Recall is the total number of correctly identified items over the expected total number of correctly identified items. Lastly, F-measure is the average of both precision and recall. Both systems also has an expert that provides the desired output from the input. This desired output would be the basis of each system whether the output that the system created is accurate or not.

2.3 Summary of Existing Systems

Table 2.1 will be showing the Summary of the explained Related Systems above.

Table 2.1: Summary of Related Systems

System	Domain	Documents	Architecture
ANNIE (Cheng et al., 2008)	Independent	Text Documents	GATE
AVATAR (Jayram et al., 2006)	Independent	Text Documents	Rule-based Techniques
SALEM (Bartolini, et al., 2007)	Dependent	Law Documents	Hobb's Architecture
Legal TrUTHS (Cheng et al., 2008)	Dependent	Criminal Law Documents	Hobb's Architecture

3 Theoretical Framework

3.1 Tokeniser

The ANNIE architecture starts with a module called the Tokeniser. This module is tasked to split text into different classifications such as numbers, types of punctuations, and types of words. Also, it uses grammar rules to provide greater flexibility and to maximise efficiency.

3.1.1 Tokeniser Rules

A tokeniser rule has two sides which is separated by a greater than (>) symbol. These sides are the left hand side (LHS) and the right hand side (RHS). The LHS is a regular expression which is matched with the input text while the RHS describes the annotations that will be added to the AnnotationSet. On the LHS, only '|' (or), '*' (0 or more occurrences), '?' (0 or 1 occurrences), and '+' (1 or more occurrences) are only allowed to be used as operators. On the other hand, the semi-colon (;) symbol is used to separate the annotations and as a result, a tokeniser rule has this format:

$$\{LHS\} > \{Annotation\ type\};\{attribute\ 1\}=\{value\ 1\};...;\{attribute\ n\}=\{value\ n\}$$

(Cunningham et al., 2009).

3.1.2 Token Types

The tokeniser considers five different token types. These are the word, number, symbol, punctuation, and SpaceToken tokens.

A token is a word token when it is a set of consecutive uppercase or lowercase letters which may include a hyphen and does not contain any other character. The word token also has an attribute called "orth" which has the corresponding values: upperInitial, allCaps, lowerCase, and mixedCaps. The upperInitial value is set when the initial letter is in uppercase and the rest are in lowercase. When a token has letters that are all in uppercase, the allCaps value is set and if everything is in lowercase, the lowercase value is set. The mixedCaps value on the other hand, is set when the token has a mixture of letters that are in uppercase and in lowercase that is not classified with the categories mentioned earlier.

A token is a number token when it is a set of consecutive digits. There are no attributes for the number token.

A symbol token is categorized into two types: currency symbol and symbol. "These are represented by any number of consecutive currency or other symbols (respectively)."

A punctuation token is categorized into three types: start_punctuation, end_punctuation, and other punctuation. Every punctuation symbol that is detected is considered a separate token.

A SpaceToken token is categorized as a space if the token is composed of pure space characters and a control if the token is composed of control characters. A set of consecutive space or control characters are considered as one SpaceToken (Cunningham et al., 2009).

3.1.3 English Tokeniser

The English Tokeniser is a processing resource which contains a tokeniser and a JAPE transducer. The transducer is tasked to make the output of the tokeniser to be readable by the English POS tagger. A specific task of the transducer would be joining together of tokens like “ ’ ”, “30”, and “s” to become “ ’30s ”. Another task would be the conversion of negative constructs like “doesn’t” in which the tokeniser outputs three tokens (“doesn”, “ ’ ”, “t”) into two tokens (“does”, “n’t”) (Cunningham et al., 2009).

3.2 Sentence Splitter

The standard ANNIE Sentence Splitter is JAPE-based and has performance issues when it comes to execution time and robustness, especially when irregular there is irregular input. Sentences that start with numbers are also not allowed by the standard sentence splitter. To address these issues, there is an alternative offered which is the RegEx sentence splitter which also segments the given text into sentences which are required by the tagger modules in ANNIE. The RegEx sentence splitter is based on regular expressions, using the default Java implementation.

The RegEx sentence splitter can be configured by three files containing regular expressions with one regular expression per line. The three files encode patterns for three kinds of splits: internal splits, external splits, and non splits. Internal splits are done when the sentence splits are part of the sentence. This happens when there are sentence ending punctuations in the sentence. External splits are sentence splits that are not part of the sentence, e.g. two consecutive new lines. Non splits are text fragments, such as full stops occurring inside abbreviations, that might be mistaken as splits but these types of splits should be ignored (Cunningham et al., 2009).

Init-time parameters

- encoding - This is the parameter that dictates the character encoding to be used while the pattern lists are read.
- externalSplitListURL - This parameter contains the URL for the file that has the list of external split patterns in the document.
- internalSplitListURL - This parameter contains the URL for the file that has the list of internal split patterns in the document.
- nonSplitListURL - This parameter contains the URL for the file that has the list of nonSplit patterns in the document. Run-time parameters
- document - This parameter contains the document to be processed.

- outputASName - This parameter dictates the name of the annotation set where the resulting Split and Sentence annotations will be created and placed.

3.3 POS Tagger

3.3.1 ANNIE's POS Tagger

This module would be using a Part-of-Speech tagger called the Hepple Tagger. This tagger associates each word or token with a part-of-speech tag. The list of tags can be seen in the Appendix A. A default lexicon and ruleset, both of which can be modified if needed, are used in the tagging process. For special cases like texts that are in all uppercase or lowercase, two additional lexicons called "lexicon_cap" and "lexicon_lower" respectively, are defined. For this to take effect, the default lexicon should be changed with the right lexicon at load time. Even if the default lexicon is replaced the default ruleset should still be used.

ANNIE's rule for unknown words is to check the suffix and try to identify what its POS is. If it still can't identify what its POS is based on the suffix, the POS tagger will tag it as a noun.

The ANNIE Part-of-Speech tagger is required to have the following parameters:

- encoding - encoding needed for the rules and lexicons
- lexiconURL - the URL of the lexicon
- rulesURL - the URL of the ruleset
- document - the document set for processing
- inputASName - the annotation set that will be used for input
- outputASName - the annotation set that will be used for output
- baseTokenAnnotationType - the annotation type for the Tokens in the document
- baseSentenceAnnotationType - the annotation type for the Sentences in the document
- outputAnnotationType - a type of annotation that will determine if POS tags will be added as category features

The process is as follows, if "inputASName" is equal to "outputASName" and the type "outputAnnotationType" is equal to the type "baseTokenAnnotationType" then features will be associated to the existing annotations of type "baseTokenAnnotationType". Otherwise, the tagger will search for an annotation with the type "outputAnnotationType" under the annotation set "outputASName" that has identical offsets with the type "baseTokenAnnotationType". It adds a new feature if successful and creates a new annotation type "outputAnnotationType" under "outputASName" annotation set otherwise (Cunningham et al., 2009).

3.3.2 Cross Reference

There are two steps in identifying the acronym and acronym definition as presented in the algorithm of Schwartz and Hearst. One is the extraction of {short form, long form} pair candidates from the text followed by the second step which is to identify the correct long form from among the candidates in the sentence that surrounds the short form. acronyms are determined by adjacency to parentheses. Two cases occur in the MEDLINE abstract and these are long form (short form) and short form (long form). If the expression inside the parentheses includes more than two words then the second pattern is assumed to have taken place and the short form will be the one to be searched for. Given the circumstances, the short forms are considered valid candidates only if they consist of at most two words with length between two to ten characters assuming one of these characters is a letter and it starts with an alphanumeric character (Schwartz & Hearst, 2003). Identifying the definition of the acronym must appear in the same sentence as the short form and should have no more than $\min(|A|+5, |A|^2)$ words wherein $|A|$ is the number of characters in the acronym (Park & Byrd, 2001).

Given the list of long form candidate words with their corresponding short form the algorithm then looks for the right subset of words. The idea is to start from the end of both the short form and long form, move right to left and try to find the shortest definition that matches the acronym. Every character in the abbreviation must match the character in the definition and the matched characters in the long form must be in the same order as the one found in the acronym. Any character in the definition is considered to match a character in the short form given that the character in the initial position of the first leftmost word in the long form matches the character at the beginning of the short form. This can be the first letter of the word connected to other words through hyphens and other non-alphanumeric characters.

3.3.3 LingPipe's POS Tagger

LingPipe's POS tagger is reliant on the documents that have been tagged manually. Lingpipe's POS tagger should be trained with the said documents. In the process of training, it creates an estimator for a Hidden Markov Model(HMM) from `HmmCharLmEstimator`, a class in the LingPipe API. The `HmmCharLmEstimator` employs a maximum posterior transition estimator and a bounded language character model emission estimator (Carpenter, 2006). A maximum a posteriori transition estimator can estimate probabilistic models, tell if the state is dynamic or the model contains hidden variables (Volkhardt, 2009). A bounded language character model emission estimators dictate the tag emissions. To create these, one must pass n-gram, total number of characters, and interpolation parameter for smoothing (Carpenter, 2006). Smoothing is the process of removing random variation and showing trends and cyclic components (*NIST/SEMATECH e-Handbook of Statistical Methods*, n.d.). The `HmmCharLmEstimator` uses a parser and it sets the handler that will be used for all the content extracted by the parser then declares the path where all the training files to be used are located. All the training files will be parsed which finishes the training producing a model file that is saved into a directory (Baldwin & Carpenter, 2004).

In using the POS tagger of LingPipe a tokenizer must be created to allow the definition of tokens. For different domains, the approximation of tokens differ. In the domain of medicine for example, the approximation of a token has the longest contiguous non-empty sequence of letter characters, numerals and apostrophes and it allows single non-whitespace characters such

as the article “a”. After the pattern identification of a token, the model file is read and casted to an HMM object which is then decoded by an object of the class HMMDecoder. The system will get the text to be tagged after reading the model. The first-best which displays the first result of tagging by the system, n-best which shows the maximum number of result of tagging by the system and confidence-based results are displayed as a result which returns all the tokens with their corresponding probable part of speech as well as its score (Cheng et al., 2008).

3.3.4 LingPipe’s Noun and Verb Chunking

LingPipe’s Noun and Verb Chunking relies on the part-of-speech tags on the tokens as well as the pattern of tags defined from a set of possible initial categories and a set of possible continuation categories. Noun chunks may start with determiners, adjectives, common nouns, or pronouns. It could then be continued with any category that may start a noun and also by adverbs or punctuation. Verb chunks on the otherhand start with auxiliaries or adverbs and continued with any of these tags or with punctuation.

The first step to chunking nouns and verbs is to tokenize the input and compute the POS tags using the HMM decoder. The tags are reviewed keeping track of the positions of the chunks and finding a start of a noun or a verb. Once a start of the noun is found the index where it starts is tracked down which is always on the first character of a token and not on a whitespace. The start index is extended one token at a time if the corresponding tag is a legal noun continuation. The end position and the overall index is kept track of. Any final punctuation will be removed. A new trimmed end chunk variable is defined (Alias-I, 2009).

3.4 Semantic Tagging

Semantic tagging is tagging an entity in the text for the purpose of classifying them. Tagged entities are then used to identify the intensions and meanings of the entity in the sentence. As stated by (Ekeklint, 2001), there are three levels to consider in tagging: words, sentences and discourse. (Ekeklint, 2001) discussed also five example categories that could address the different levels namely, sense-tagging, feature tagging, compositional semantic representation, dialogue-act tagging, and document tagging.

Sense-tagging is the tagging of entities or tokens with their corresponding sense in the text. This type of category is usually used to tag words. An example where sense-tagging could be used is with the word “support”. “Support” in these sentences: “My support goes to the bulls” and “The support of the building is strong”, is not used in the same sense. Additionally, tools like POS tagger, dictionary definition, domain codes, selectional preferences, and collocates, are used by sense-tagging to lessen disambiguation and improve its performance. The POS tagger is used to lessen the disambiguation of some English words that could have two or more parts of speech. An example of this is the word “record” where it could be a noun or a verb depending on the usage it is on the sentence. In the sentence, “The runner has beaten the all-time record”, “record” is used as a noun while in the sentence, “The officials record the time” is used as a verb. Second, dictionary definitions are used in measuring the semantic closeness. In addition, it also computes the overlap of definitions for a sentence and by this, proposes the disambiguation of sentences. Another tool that is used for semantic disambiguation is collocation. Collocations are phrases that are seen together more than expected given an estimate of

how frequent each token are seen together (Cheng et al., 2008).

Feature tagging is using more abstract categories or features for tagging. It is used to identify semantic relations between lexicalised words (Ekeklint, 2001).

Compositional semantic representation is one of the most common methods in representing the semantics of sentences. It is the process of tagging an entity or a token reflecting their compositional properties. Dialogue act-tagging is the tagging of the dialogue act in the series of dialogue. It tags them according to which type of classification it belongs. An example of a classification could be sorted among the types of sentence that the dialogue act fits.

Document-tagging is classifying the document into a category such as business and sports.

3.5 Named Entity Recognizer

Named Entity Recognition or NER, is a process wherein strings are recognized into predefined categories like names, organizations, places and others. Categories may vary depending on the domain.

The Baseline Information Extraction's or Balie's named entity recognition makes use of a lexicon as a basis for recognizing strings. The process involves by checking the lexicon and listing all the categories wherein the string was found. So if a string fits in multiple categories, all the categories will be associated to that string (Nadeau, 2005).

LingPipe's named entity recognition works in three ways. It can either use dictionary matching, train a statistical model, or use rule-based named entity detection.

LingPipe's dictionary matching is done by matching a token directly to the token in the dictionary list. Using statistical and rule-based approach on the other hand, would be difficult since there are words like "360Blu blu-ray player" which is hard to recognize. LingPipe's dictionary matching has feature allowing users to create a dictionary containing words which are user defined. Completely different from dictionary matching, rule-based entity detection uses patterns to match a token to a certain category. The statistical model on the other hand, has three examples namely, first-based named entity chunking (FNEC), N-best entity chunking (NNEC), and confidence named entity chunking (CNEC). Models are produced by training the data with the chosen documents. FNEC returns an answer by reading the model which effectively results in a Chunker and with this, it will be able to return a character sequence which is a found entity. NNEC arrives in an answer by using nbest method of the "NBestChunker". This returns results which contains the score and the character sequence of the found entity. The third one, CNEC returns an answer by ranking of confidence. Results are the confidence score (two raised to the score of the chunk), start and end index of the entity, and type of the entity, and the entity itself (Baldwin & Carpenter, 2004).

3.6 Coreference Resolution

There are a number of solutions for anaphora resolution. These solutions can either be rule-based or even adopt machine learning approaches. Discussed in this section are two pronoun

resolution algorithms in which the approach used is rule-based.

3.6.1 Ruslan Mitkov

As most approaches to anaphora resolution rely a lot on linguistic and domain knowledge which is time consuming, this approach however, presents a robust, knowledge-poor approach in resolving pronoun referencing issues. The approach uses inputs that are pre-processed by a part-of-speech. By checking the input against agreement and antecedent indicators, candidates are found and assigned scores in which the candidate with the highest score is selected as the antecedent.

Aside from using a part-of-speech tagger, this approach also uses noun phrase rules and runs by the help of antecedent indicators. Using the input from the part-of-speech tagger, it identifies the noun phrases within 2 sentences back. Then it checks the gender and number of the noun phrase if it agrees with the anaphor and applies the genre-specific antecedent indicators (definiteness, givenness, indicating verbs, lexical reiteration, section heading preference, "non-prepositional" noun phrases) to the remaining candidates and at the same time, scores are assigned. The noun phrase with the highest score is the assigned antecedent (Mitkov, 1998).

3.6.2 Lappin Leass'

The Lappin and Leass' algorithm also known as RAP (Resolution of Anaphora Procedure) identifies both intrasentential and intersentential antecedents of pronouns in a given text. This approach relies on the measures of salience obtained from the syntactic structure and a model for identifying the antecedent from a list of candidates.

RAP has an intrasentential syntactic filter for removing anaphoric dependence of a pronoun on a noun phrase based on syntax and a morphological filter for non-agreement of person, number, or gender features. RAP also has a method for identifying pleonastic pronouns, or meaningless pronouns. Another component is the anaphor binding algorithm that identifies possible antecedent binder of a lexical anaphor within the same sentence. Since it relies on the measure of salience, the algorithm has a procedure for assigning values to a number of salience parameters such as grammatical role, parallelism of grammatical roles, frequency of mention, proximity, and sentence recency for a noun phrase. Also, there is a procedure for knowing if noun phrases are an equivalence class for which a global salience value is calculated as the sum of the salience values of its elements. Lastly, RAP has a decision procedure for choosing the best element in a list of candidate antecedents for a pronoun (Lappin & Leass, 1994).

3.7 Hidden Markov Model

The Hidden Markov Model, to be used in the POS Tagger, is composed of a finite set of states. Each state is associated with a generally multidimensional probability distribution. Transitions among the states is dependent on a set of probabilities called transition probabilities. An outcome or observation can be generated in a particular state according to the associated prob-

ability distribution (Warakagoda, 1998). Figure 3.1 shows the components of the HMM. Each

A hidden Markov model (HMM) is a triple (π, A, B) .

$\Pi = (\pi_i)$	the vector of the initial state probabilities;	
$A = (a_{ij})$	the state transition matrix;	$Pr(x_{i_t} x_{j_{t-1}})$
$B = (b_{ij})$	the confusion matrix;	$Pr(y_i x_j)$

Figure 3.1: Hidden Markov Model Components

probability in the matrices is time independent which means the matrices do not change in time as the system evolves. This is one of the most unrealistic assumptions of Markov models about real processes (Boyle, n.d.).

3.8 Tribayes

Tribayes applies the trigram method first while determining whether all the words in the confusion set would have the same tag if it's substituted into the target sentence. If it doesn't, tribayes would accept the answer provided by the trigrams however if it does, it would apply the Bayes method. It is necessary that Bayes, in this role would be trained on a subset of the examples that would be used for the training of a stand-alone version of bayes. If both trigrams and bayes make uninformed decisions meaning that the decisions are only based on the priors, tribayes would accept the result from the bayes method (Golding & Schabes, 1996).

3.9 Modified Three Way Match Method

The algorithm described by Elmi and Evens makes the minimum edit distance or the number of deletions insertions and replacements given an incorrect string S and the word from the lexicon W to transform S to W. Error types can be classified into four: reversed order, missing character, added character and character substitution. Elmi and Evens' algorithm extended the edit distance by assigning weights to each correction taking into account the position of the character in error.

In character substitution, if the erroneous character is a neighbouring key of the character on the keyboard or if the character has a similar sound to that of the substituted character, the error weight is reduced by 10%. The string comparison step of their algorithm was based on a system developed by Lee and Evens (1992). If the character at the location n of S does not

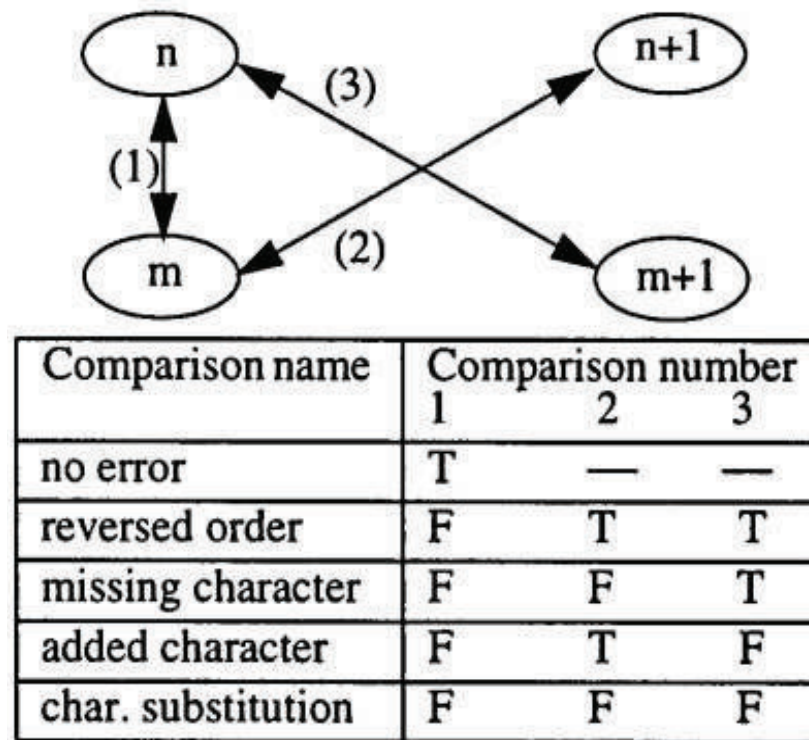


Figure 3.2: Modified Three Way Match and the order of Matching

match the character location m of W , an error is detected and two other comparisons will be made. This algorithm is called the three way match (3wm) as shown on Figure 3.2.

For an added character error when a 3wm detects an added character error and $\text{char}(n+1) = \text{char}(m+1)$ and $\text{char}(n+2) \neq \text{char}(m+1)$ the error to character substitution type is changed. For a missing character error, if the 3wm detects a missing character and $\text{char}(n+1) = \text{char}(m+1)$ the algorithm checks further conditions. If $\text{char}(n+1) \neq \text{char}(m+2)$ or $\text{char}(n+2) = \text{char}(m+2)$ the algorithm changes the error to character substitution. If a 3wm detects a reverse order error and $\text{char}(n+2) \neq \text{char}(m+2)$ the error will be changed to a missing character error or an added character error. This is dependent on the result of further conditions to be fulfilled. If $\text{char}(n+1) = \text{char}(m+2)$ the algorithm will insert $\text{char}(m)$ at the location on of the misspelled word. However if $\text{char}(n+2) = \text{char}(m+1)$ $\text{char}(n)$ is dropped. A novel thing that Elmi and Evens added to the 3wm is to handle two mismatching characters which is actually problematic for many spelling corrector systems. When comparing S with W , the algorithm of Elmi and Evens partition them as $S = xuz$ and $W = xvz$ where x is the initial segment, z is the tail segment, and u and v are the error segments. The initial segment is selected for the first step. This segment could possibly be empty in an instance where S and W do not match. It is unlikely that S will be equivalent to W , in this case the segment will contain the whole word. The second step to this algorithm is the tail will be selected and can be empty if the last characters of S and W are different. Error segments are the remaining characters of the two words (Elmi & Evens, 1998).

3.10 MySQL

MySQL is the most popular open source SQL database management system developed, distributed and supported by Sun Microsystems, Inc. It was written in C and C++, it was tested on different compilers and it works on many different platforms. MySQL also provides transactional and non-transactional storage engines. It supports many data types, such as signed or unsigned, fixed length string types and variable-length string types. It gives full support on almost all SQL statements namely clause of query, group functions, statements that retrieve information and statements that shows how it does a query. One can also refer to another table from another database in the statement. As for security, MySQL provides a flexible and secure password system and it also allows verification from the host. MySQL can support a large amount of database, Sun Microsystems uses it and it contains 50 million records. Programs that will be using MySQL could be written in many languages because it supports APIs for C, C++, Eiffel, Java, Perl, PHP, Python, Ruby and Tcl. MySQL also gives full support to different character sets such as latin, german, big5, ujis. MySQL includes command line programs and graphical programs, it also supports SQL statements which checks, optimizes and repairs tables (SunMicrosystems, 2008).

3.11 MyDMS

MyDMS is an open-source document management system (DMS) made up of both PHP and MySQL and published under the GNU General Public License. MyDMS allows the storing of any kind of binary data like a file-system. However, unlike a file-system MyDMS can have meta-data attached to the documents in MyDMS. Old versions of documents are saved and can be retrieved if the user wants to do so. MyDMS also provides easy access when users want to find files through tables of contents, indexes and full-text searches and sort the results using a certain criteria. MyDMS provides a way for users to upload files which can either be submitted or published (viewable by others) through web-interface, create folders to group the user's documents, edit the document and folder properties online, provide detailed information on uploaded documents or meta-data, lock and unlock documents, update documents while saving old versions at the same time, set expiration-dates for documents, download documents or view them online on the user's browser, control access via detailed access control lists, choose language and theme on login and manage users and groups. MyDMS also offers individual icons for different mime-types, notifications about new, updated or expired documents through their e-mails, a search engine, multi-language support, an intuitive user interface and a template system. It is also compatible with most browsers, converts documents to HTML and supports multiple databases through the use of ADOdb which is a database abstraction library (TrileXLabs, 2008).

3.12 Evaluation Metrics

This section would be discussing about the evaluation metrics that would be used during evaluation of the system.

3.12.1 Precision

The measure would be the percentage of the number of correctly identified items over the total number of items. The higher the output percentage the better the accuracy of the system (Makhoul, 1999).

The formula of precision is:

$$\text{Precision} = \frac{|\{relevant\ information\} \cap \{retrieved\ information\}|}{|\{retrieved\ information\}|} \quad (1)$$

3.12.2 Recall

The measure would be the percentage of the number of correctly identified objects over the expected number of correctly identified objects. The higher the output percentage the higher the chance in getting all correctly identified objects (Makhoul, 1999).

The formula of recall is:

$$\text{Recall} = \frac{|\{relevant\ information\} \cap \{retrieved\ information\}|}{|\{relevant\ information\}|} \quad (2)$$

3.12.3 F-measure

The measure would be the average of the measure of precision and recall which would be of help to make sure that the system would be reaching 100% in both precision and recall (Jones, 2006).

The formula of F-measure is:

$$F_{\alpha} = \frac{(1 + \alpha) * precision * recall}{((\alpha * precision) + recall)} \quad (3)$$

3.13 Resources

Resources are necessary in the information extraction process. The resources that are needed can include a training corpus, models and lexicon. The tools that will be used in the system will be the one to determine which resources should be used.

3.13.1 Training Corpus

A training corpus is a set of documents. The corpus is annotated with metadata so that it can be used as training documents for generating models or rules for certain tools. An example of a tool that uses a training corpus is LingPipe's POS tagger. LingPipe's POS tagger relies on manually tagged documents therefore, it is trained with a specific set of documents.

3.13.2 Lexicon

A lexicon will be the system's vocabulary. It will contain the words and will be classified into certain categories. The words could be generic and some words could be something more specific. Examples of more specific words would be names of companies and names of persons. In addition, a lexicon may contain additional information about the words which could be important. An example of this is a lexicon that contains the original words and its corresponding acronyms such as De La Salle University that has the acronym DLSU. Another example would be lexicons of female and male names which serve as a lookup for a tool like the Named Entity Recognition.

3.13.3 Models

According to Cheng et al (2007), "a model or a rule set is a set of assumptions/approximations about how the system works" (p.3-23). A range of models can be generated by current tools given a training corpus. Given an input, the purpose of a model is to produce an output that will match to a certain task (e.g. Named Entity Recognition and POS tagging). There are several kinds of models in information extraction. And for each language, genre, or training corpus, available tools like LingPipe would be able to produce a model based on a specific task.

4 System Objectives

This chapter includes the system's overview, general and specific objectives, and scope and limitations. The main functions and components to be included in the proposed system are also defined in this chapter.

4.1 System Overview

IE for eLegislation is an information extraction system that extracts important information from different types of documents found in BRC. The system works with an existing DMS that contains the electronic versions of the documents. These electronic versions of the documents serves as the input for the system. Each type of document is associated with a template containing pre-defined fields. The fields of these templates is filled by the system with the corresponding values extracted from the corresponding document discarding the irrelevant and non-important words inside the document in the process of filling up the templates. The output of the system are completed templates that will be stored as records in the system's database.

The system has a front-end search module that enables users to enter keywords into multiple fields. As compared to the simple keyword search wherein you only have one field for searching, multiple field searching or advanced searching will result to a more narrowed down search output and a more efficient execution time for searching. In addition, with the help of synonym detection, the system is able to return probable related documents to the interest of the user.

4.2 General Objective

To extract information from the given Blue Ribbon Committee documents and assign the extracted information to their corresponding fields in the template.

4.3 Specific Objectives

To be able to:

1. Open and read Microsoft Word documents or text files from the document management system and divide them into partitions;
2. Preprocess the document such as identify sentence boundaries and apply POS Tagging;
3. Identify and normalize basic entities in order to comply with the designated standard format;
4. Discard information that are most likely irrelevant;
5. Use anaphora resolution to resolve issues in key entities;

6. Construct a database consisting of tables that will hold specific filled-up templates which contains the information from the input document;
7. Evaluate and provide a report of the performance of the information extraction system;
8. Detect synonyms of query and return probable related documents to the interest of the user;

4.4 Scope and Limitations

The research assumes that there is an existing Document Management System that manages the text file or word document. In effect, these text files or Microsoft Word documents will be used as input by the proposed system. MyDMS is one of the proposed Document Management System that was used to connect to the IE system. Currently, there is already a prototype of the DMS implemented by a resource person outside of the group.

The Blue Ribbon Committee documents that the system processes are the hearing highlights, hearing invitation, senate memorandum, documented evidence, complaint endorsement, referrals, notice of hearing and committee reports. These are stored as Microsoft Word documents or as text files in the DMS and these documents should be cleaned to make sure that it is already grammatically and morphologically correct. These are the documents that are chosen because the language used in these documents are purely English except for the committee report, wherein Tagalog words are found more in dialogues which will not be processed. To recognize the type of input document, the Blue Ribbon Committee should follow the prescribed filename convention prescribed by the research.

The system identifies the type of Blue Ribbon Committee document depending on the specified filename and with that filename the correct template would be identified and used. If the specific filename is not followed then the system would not accept it as input. The text files are divided into different partitions and from the different partitions produced, the system finds the boundaries per sentence. The words are then tagged by the system to its corresponding part of speech (POS).

Since there are many kinds of Blue Ribbon Committee documents, there is a possibility that some words of the documents would not be recognized and processed properly by the system. Therefore, the proposed system has heuristics to deal with these unrecognized words found in the documents. There is also the possibility that there would be noise inside the document, such as misspelled words, punctuation errors, images, OCR errors, evidences, punishments and dialogues. The evidences, punishments and dialogues are still subject to change depending on the preference of the Blue Ribbon Committee. The system would know the words are misspelled if there is a word "[SIC]" after the corresponding word. All words, correctly spelled or not, are still handled. When the user searches using that certain field and inputting the correct spelling of the word the system would not show the result with the misspelled word. For punctuation errors, periods instead of commas, the system divides the sentence into two different sentences, for this situation the system checks whether the next letter after the punctuation is in capital or not. If it's in capital form then it will be divided but if the next letter is not a capital letter then the system continues until it reaches another punctuation. Images, evidences, punishments and dialogues are all removed from the document. OCR errors are not handled by the system, it is assumed that once that the document is published by the Blue Ribbon Committee the OCR errors would no longer be present.

The system looks at different entities in the documents, in which these entities would be normalized for it to have a standard format that will be saved in the database. The protocol is one table for each different type of document.

The database of lexicons used by the modules will be populated by data from different sources in respect to type of data. The data of Male Names, Female Names, Pronouns, Noun Phrases tables are from a website that hosts a list of native names from the Philippines. On the other hand, the data of Organizations table are from a publication of the Department of Trade and Industry for single-owned businesses and a publication of the Philippine Securities and Exchange Commission for business owned by two or more people. Additionally, some of the data are provided by the Blue Ribbon Committee.

The compound words are joined together, an example would be "University of the Philippines" wherein the phrase should be counted as a whole. Lexical disambiguities are handled by the system. The words are then checked if they are referring to the same meaning or not. As for the referencing issues, the proposed system uses anaphora resolution to resolve this problem.

The system can identify the pre-defined template that will best suit the input text file. The information extracted from the input text are then used to fill up the template and finally, be placed in the database. Also, the information extracted is based on the type of document extracted. The fields are based on the preference of the resource people from Blue Ribbon Committee.

There is a database which can be a third-party webserver or one of Blue Ribbon Committee's own in which each document type will have a specific table. The tables have fields or attributes corresponding to the document's pre-defined templates.

The Goal Standard used for evaluation was given by resource persons. The resource persons provided filled-up templates and it will serve as a reference if the output of the system is accurate or not. Details of the evaluation process is discussed in Section 4.5.7.

The system returns relevant results using synonym-detection by using an existing tool, Wordnet. This is done once the search engine invokes a query. The information extraction process is done per document published in the DMS.

The system includes in its search, synonyms of the terms specified by users. In effect, related documents would also be retrieved. This is done using Wordnet, an existing synonym detection tool.

List of Assumptions and Limitations:

- The Information Extraction System produced by the research would be accepting text files that contain pure text. Non-text sections would be removed and not processed.
- The system would be focusing on documents that are provided by the Blue Ribbon Committee, particularly hearing highlights, hearing invitation, senate memorandum, documented evidence, complaint endorsement, referrals, notice of hearing and committee reports.
- The filename convention prescribed by the research should be followed by the Blue Ribbon Committee when storing files into the DMS.

- The Information Extraction System will use existing tools such as Tokenizer, Sentence Splitter, POS Tagger, Named Entity Recognizer and Preparser.
- The three stakeholders are BRC, NGOs and citizenry and each of the stakeholders have one resource person as a representative, will label some sample documents and will be used as basis on the creation of pre-defined templates. The labeled sample documents will be used as basis on the automated evaluation of the accuracy of the system.
- The database filled by the IE system will be used by the front-end search module.

4.5 System Architecture

Before the information extraction modules start, the system would first determine the type of document by looking at the filename of the input document. The filename would need to specify the type of document to be processed for the system to know the template and rules to be invoked.

4.5.1 Pre-processor

The pre-processor will have a single document as an input. This module is tasked to clean the input by remove noise such as styles present in some documents. The result would be a document that does not contain any images, graphics, headers, or footers.

4.5.1.1 Unicode Tokeniser

The Unicode Tokeniser accepts as input the file from the pre-processor. It divides the document into simple tokens like numbers, punctuations and words. These tokens are then associated with types such as word, number, symbol, punctuation, and space. The output of this module is a list of tokens. For example, given the sentence "I have 25 pieces of paper" the module will output "I_word" "have_word" "25_number" "pieces_word" "of_word" "paper_word". The research plans to use ANNIE's Unicode tokenizer module.

4.5.1.2 Sentence Splitter

The Sentence Splitter module basically accepts as input the list of tokens and annotation lists as input. This module uses the list of abbreviations to distinguish a simple period from an abbreviation period and the list of tokens to form a sentence. The final output would be a list of sentences. For example, given the sentences "I have a dog. My dog's name is fluffy" the system will output "I have a dog" and "My dog's name is fluffy" as two different entities. The research plans to use ANNIE's alternative sentence splitter which is the RegEx sentence splitter implemented in JAVA.

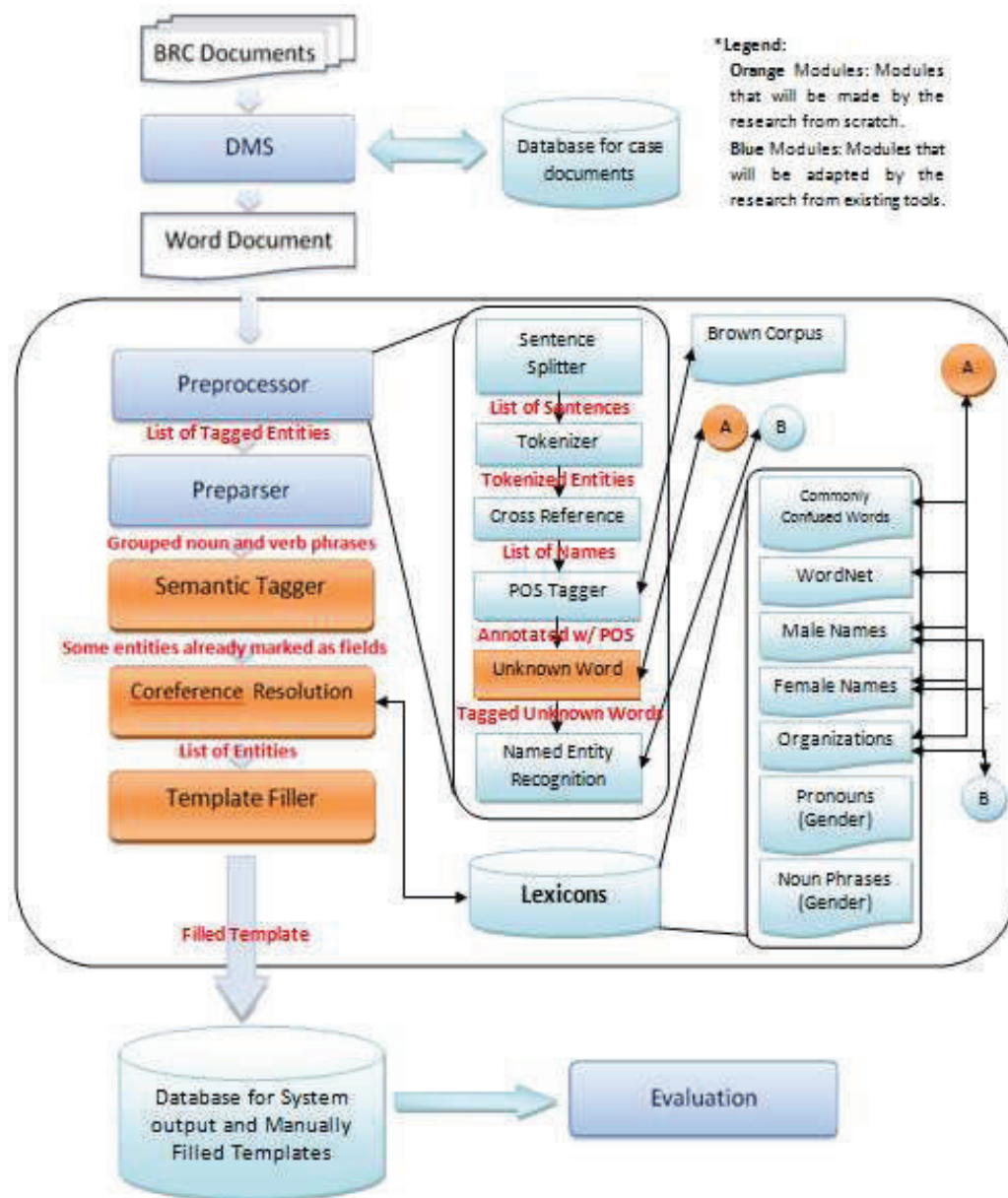


Figure 4.1: System Architecture

4.5.1.3 Cross Reference

The cross-reference module builds up a lookup table of synonymous informal words and formal proper nouns. The input would be the list of tokenized sentences as well as the output of the preparser. It does not require any additional data other than that. This module has been designed in order for the output to be of use at the front-end side of the system. (e.g. given

Department of Agriculture(DA) the module would have to add department of agriculture(DA) to the database of informal words and formal words wherein DA is the informal word and Department of Agriculture is the formal word) This is only applicable to organizations mentioned in documents as well as names(e.g. given Jocelyn "Joc-Joc" Bolante, the module must be able to detect Joc-Joc as the informal word and Jocelyn Bolante as the formal word synonymous to it). As an additional function of this module, this also automatically adds to the dictionary of proper nouns relevant to the domain to be referred to by the NER module.

4.5.1.4 POS Tagger

The POS Tagger module accepts as input the list of sentences. Processes in this module involve the tagging of tokens into its corresponding Part- of-speech, whether it's a noun, adjective, adverb, verb or others. The output will be a list of tokens each with their corresponding part-of-speech tag. The research is considering on using LingPipe's tool for POS tagging. The sample output would be "the_article" "sad_adjective" "clown_noun".

4.5.1.5 Unknown Word

The Unknown Words module tries to tag the unknown words according to its proper part of speech. The research plan to use a Filipino dictionary since it's probable to have the Filipino words marked as unknown. For other unknown words the research plan to adapt ANNIE's algorithm to give a wise guess on what the unknown word's part of speech is. For example, the words that end with ness will be tagged as nouns while those that end with ly will be tagged as adverbs words that can't be identified such as itak would be considered as a noun. Words with [SIC] succeeding it will be considered as unknown words. These however will have more special rules. The system will check first if the word or phrase being processed is proper this is done by looking if the word's initial letter is capitalized. If it is a word that exists in the lexicon, the system would apply the three way method. However if the word does not exist in the lexicon it would be using context sensitive spelling correction using trigrams and feature-based methods.

4.5.1.6 Named Entity Recognition

This module takes as input the tokens from the POS Tagger so that nouns are already identified. This module will be focusing on proper nouns. The research are considering on using LingPipe's named entity recognition tool. Among the three ways, dictionary matching, statistical and rule-based named entity recognition, the statistical named entity recognition was chosen because by using the dictionary based approach, huge amount of resources will be needed in order to classify entities. Another reason is that using the rule based approach would require tedious and time consuming work in constructing the rules. Basic named entities are names of persons, organizations, and locations which are listed in tables of a database.

4.5.2 Filter

The Filter module accepts as input the list of tokens from the semantic tagger. It then filters the tokens that are irrelevant from the list to make it easier for the succeeding modules to process. Those irrelevant tokens are the information that are not important in the document determined by the resource person. The output of this module is the list of tokens without irrelevant. An example of irrelevant information to be filtered out are dialogues that could be found in committee reports.

4.5.3 Preparser

The phrase chunker module accepts as input the set of tokens tagged by the POS tagger. This module groups words to form noun phrases and verb phrases. The group plans to use LingPipes Noun and Verb chunker to determine noun phrases and verb phrases. This module will be useful for the coreference resolution module as well as the semantic tagger module afterwards. For example with an input of John Smith will be seen the next day. John Smith will be chunked as one entity tagged as a proper noun phrase and the next day will also be chunked as one entity as a noun phrase and will be seen the next day will be chunked together and tagged as a verb phrase.

4.5.4 Semantic Tagger

The next module would be the semantic tagger module. This module will remove the ambiguities from the phrases or words or tokens. This module will also identify the template that it will use in identifying which tokens are most likely the candidates for the given fields in a certain template by referring to the document type parameter that will be passed by earlier modules. This identifying of candidates would be possible with the help of rules and pattern matching heuristics that corresponds to a certain template. This module will also be developed by the research. An example would be the list with entities "Richard Gordon", "June 30, 2009" and "General Luna St., Intramuros Manila". For a certain template such as the Hearing Invitation, "Richard Gordon" would be tagged as the "Receiver", "June 30, 2009" would be tagged as "Date Sent" and "General Luna St., Intramuros Manila" would be tagged as "Address". However, with the Hearing Highlights template, "Richard Gordon" would be tagged as the "Receiver", "June 30, 2009" would be tagged as "Date Sent" and "General Luna St., Intramuros Manila" would be ignored.

4.5.5 Coreference Resolution

This module is used to solve referencing issues for pronouns and noun phrases. The research plan on adopting the Ruslan Mitkov approach on solving this problem. To be able to do this, the tokens should already be annotated with their part of speech. Lexicons of female and male names should be available to be able to check the gender of the noun phrase or pronoun. The lexicons will be populated from internet sites found containing the names. The list of organizations and companies on the other hand, will be acquired from the security exchange commission (multiple owners) and DTI websites (single owner).

4.5.6 Template Filler

The last module of the information extraction system is the template filler. The fields of a certain template are different for each kind of document (see Appendix C). This module will identify the template that it is going to use by referring to the document type parameter by earlier modules. When the type of template is identified, it will fill up the fields of the template. All extracted information that is entered in the template is normalized to a specific format. This is also the module that would output the completed template that will be saved in a database. It is important to note that the template is not yet finalized and may still be changed by the resource persons.

4.5.7 Evaluation

The evaluation module is the module that will be accepting two templates as input. The first would be the output, filled-up template, of the system and the other one is the desired filled-up template of the resource person. The two templates would then be compared to check the accuracy of the system. The output of this module would then be a report of the percentage of the output of the system, if it is extracting the correct information from the input document. The documents would be divided into two kinds, namely as training data and test data. Many systems are doing that in 100 documents, around 90 would be used as training data and the remaining 10 would be used as test data (Yamada, Sakano, Yasumura, & Uehara, 2006). The remaining 10 of the documents would be used to compare whether the information extracted from the document is matching the filled-up template prepared by the resource person. To determine the accuracy of the output, the system would be using measurements such as precision, recall and f-measure. Precision is the percentage of correctly identified items in the output of the total number of items identified. Recall is the percentage of correctly identified items in the output of the total number of expected correct output. F-measure is the average of both Precision and Recall, which balances the percentage of both measurements.

4.5.8 Search

The search module of the system would be term-based just like other search engines. The difference would only be that the user can choose where to put that specific term in the search engine. The system would let the user choose what kind of document is to be searched. After choosing, the user would be seeing that there would be fields shown in the window wherein they could fill up with the term that they would be searching for. For example, the user chooses Hearing Invitation as the kind of document, then the system would be showing the list of fields that are considered as important in the document, the user would then input in one or more of the fields, after filling it up and clicking search the system would then use the terms entered by the user to search in the database of filled-up templates. Only the fields that has search inputs would be used to search in the database. This kind of search would make the searching easier because it would not need to search the whole document.

4.6 Hardware and Software Specification

4.6.1 Development

4.6.1.1 Minimum Software Requirements

- MyDMS
- PHP 4.0
- MySQL 4
- ADODB database abstraction library
- Windows XP
- Apache Server

4.6.1.2 Minimum Hardware Requirements

- 1GB RAM
- High-Speed Internet Connection
- Server

4.6.2 Deployment

4.6.1.1 Minimum Software Requirements

- MyDMS
- PHP 4.0
- MySQL 4
- ADODB database abstraction library
- Windows XP
- Apache Server

4.6.1.2 Minimum Hardware Requirements

- 1GB RAM
- High-Speed Internet Connection
- Server

5 Design and Implementation Issues

This chapter would be discussing the implementation of the modules created by the research. This chapter would also be discussing the issues encountered by the research in the process.

5.1 Introduction

The Information Extraction System for eLegislation is composed of 11 modules all in all. In those 10 modules, there consists 7 minor modules and 3 major modules. The 6 minor modules are Sentence Splitter, Tokenizer, CrossReference, POS Tagger, Unknown Word, Named Entity Recognition and Preparser and the 3 major modules would be Semantic Tagger, CoReference Resolution and Template Filler. In addition to the 11 modules above, the research implemented external modules to address the needs of the Blue Ribbon Committee, these external modules were the Search module for the Front-End search engine and Evaluation module to evaluate the system. Other than these modules, there are also other modules or packages that the research had used in order to connect to the database and to open documents. In addition, lexicons were also used in order to train certain modules. The system was designed to handle 8 kinds of documents, namely Agenda, Committee Report, Hearing Highlight, Hearing Invitation, Notice of Hearing, Order, Scenario and Subpoena.

5.2 Data Gathering

This subchapter would be explaining how the research gathered the necessary data from the resource persons.

5.2.1 Documents

In order to reduce the tediousness in inputting test data into the DMS to be manipulated the information extraction system this research aims to develop, Optical Character Recognition(OCR) will be used. OCR is a scanning and comparison technique used in order to identify printed text or numerical data whether handwritten or machine produced hard copy of a document. It compares the shapes found in the scanned document to those stored in the software library. The software would then try to identify the words using character proximity and rebuild the original page layout in soft copy. It is suggested that a sharp and clear scan of a document would be used for better results ("OCR(Technology)", 2010). It is inevitable for OCR processes to contain errors therefore affecting the further stages in the accuracy of the output given the input that has gone through OCR. (Lopresti, 2008).

The research were given hard copies of documents that were fed to a scanner. It would be easier if the scanner that would be used can handle stacked documents. In this case the research used Canon MX310 and the OCR program that comes along with its driver which is MP Navigator EX. the research have also sampled a standalone free OCR called Simple OCR. The results however were noisier than that produced by the OCR component of the Canon MX310 driver. the research resulted to, after scanning and compiling the stack of documents into a pdf format and invoking the OCR option, copying and pasting to Microsoft Word what the result

was and manually looking for noise that were not handled in the OCR. The research did not dwell on this matter further for it was assumed in the scope and limitations based on interviews done with the resource persons in the Blue Ribbon Committee that it is their duty to check what the contents of the documents to be uploaded are.

Other than the hard copies being given to the research, the research also asked for soft copies of the documents to be used as input in the system created. These soft copies of the document were asked for because the hard copies of the documents were not that clear and most of the documents had noise in them, such as stamps and signatures of the receivers. The resource person had a hard time giving the research the soft copies because they do not keep soft copies of the documents that they had printed. They would just modify an old one in their archive, and input the necessary details for the current case. They just use "Save As" and they continuously just update it for new cases. The work around for this would be that the research manually encoded the input for the system based on the hard copies given by the resource person.

5.2.2 Templates

This subsection would be explaining how the research gathered the templates of the system.

5.2.2.1 NGO

For the NGO templates, the research had interviewed some people in order to know the templates needed by them, most of them are professors and some of them are affiliated with the Blue Ribbon Committee which was contacted by the resource person from the Blue Ribbon Committee. Some of the NGOs that the research had interviewed would be Sir Jayson Hecita referred by the La Salle Institute for Governance who is currently in Florida, Chair of the Political Science Department Dr. Eric Vincent C. Batalla and Sir Rico one of the political student leaders in De La Salle University. The research photocopied copies of the documents and highlighted the fields that the Blue Ribbon Committee had asked for. The NGOs checked whether they wanted to add more fields or they would like to remove some of the fields from the template. After consulting with the NGOs, the research would send the desired template of the NGOs to the resource person in the Blue Ribbon Committee. The resource person would be the one that would be validating the template, she will be the one to check whether the fields asked by the NGOs are allowed or not. Overall, the fields of the NGOs and BRC were the same, the only additional feature that the NGOs wanted would be the cross referencing in the search module, they would want to see synonyms and referencing when a keyword is searched from the front-end module.

5.2.2.2 Blue Ribbon Committee

The sample templates from the Blue Ribbon Committee were given to us personally by one of the personnel in the BRC. They have stated the important information that they would like to see as the output of the system. These templates were then re-written in table manner for rechecking, the re-checking process would be that these tables created would be passed on to the personnel from the BRC so that he/she could check whether the fields are complete

or not. When the templates are already finalized, the system would be basing the output of the system from the templates that had been created. After getting the initial template from the Blue Ribbon Committee, the research still had constant communication with the resource person, so whenever the resource person wanted to add another field in the templates of the system, then the research would immediately be updated and the necessary changes would be made in order to accommodate the needs of the Blue Ribbon Committee.

5.3 Modules

5.3.1 Preprocessor

5.3.1.1 Sentence Splitter

The sentence splitter is the module that splits the input text into sentences. The input for this particular module is the word document and then the corresponding output would be the list of the sentences found in the document.

The basic sentence boundaries that could be found in the documents are periods (.), question marks (?), and exclamation points (!). These sentence boundaries are being searched for in the document and when it reaches that certain sentence boundary, then a sentence is found. The sentence is then stored into the list of sentences. After finding the sentence boundaries, the function would then look at the next value. The action to be done next, if it would be added to the current sentence or it would create another sentence and save the previous sentence in the LinkedList. It will continuously add to the LinkedList of Sentences inside the function until it reaches the end of the document.

Table 5.1: Sentence Splitter 1

Condition	Action
if character equals space	tempString plus space; size++;
If character equals to tab	tempString plus tab; size++;
if character equals period or question mark or exclamation point or next line or return	Check if tempString could be found in abbreviations list
if found	Sentence plus tempString
If not found	Refer to Table 5.2

Table 5.2: Sentence Splitter 2

Condition	Action
if the value after the sentence boundary is a lower case letter	sentence plus tempString
if the value after the sentence boundary is an upper case letter	sentence plus tempString add the sentence to the sentenceExtracted list of sentences
Continued on next page	

Table 5.2 – continued from previous page

Condition	Action
if the value before the sentence boundary is a number and the value after the sentence boundary is also a number	sentence plus tempString
if the value after the sentence boundary is a number	sentence plus tempString add the sentence to the sentenceExtracted list of sentences
if the value after the sentence boundary is a letter and the value before the sentence boundary is a number	sentence plus tempString

There are certain issues that were encountered during the process of creating this module. One of which is concerned with the abbreviations that may appear in the document. In this case, the system would be considering it as a sentence boundary, but the research devised a work-around that required the system to maintain a list containing abbreviations which served as a look-up table to see if whether the sentence boundary is part of an abbreviation or not. Another issue concerns with enumerated lists. The issue here is that there were also sentence boundaries that could be found. To address the issue, the system would be checking whether the value before the sentence boundary is a number, if its a number then the system would be continuing the process of searching for a valid sentence boundary.

5.3.1.2 Tokenizer

The tokenizer is the module responsible for splitting the text into individual words, numbers and symbols. It uses white spaces as its delimiter for finding the individual tokens. The system adapted the algorithm of ANNIE's tokenizer. This modules output would be a list of tokens which are tagged with their corresponding token type. The code below indicates on how the research used and implemented this module side by side with ANNIE's algorithm in the tokenizer.

Table 5.3: Tokenizer 1

Condition	Action
else if Character token[i] is a digit	start = i; i = checks up to what length of token is a valid number or order; returns i is to continue. t sets token substring from start to i to token attribute. t sets Number to kind attribute. t is added to tokens.
else character token[i] is a special character	t sets token[i] to token attribute. t = checks the kind of special character is token[i]; returns the token t. t is added to tokens

Table 5.4: Tokenizer 2

Condition	Action
If length of token is greater than 1	i = check if words orth value; returns i is to continue. t sets token substring from start to i to token attribute. t sets Word to kind attribute. t sets the orth to orth attribute. t is added to tokens.
else if Character token[i] is in uppercase	t sets token[i] to token attribute. t sets Word to kind attribute. t sets allCaps to orth attribute. t is added to tokens.
else character token[i] is in lowercase	t sets token[i] to token attribute. t sets Word to kind attribute. t sets lowercase to orth attribute. t is added to tokens.

The Tokenizer splits the input into tokens with respect to white space tokens and according to word, number, and symbol tokens. There were no issues encountered implementing this module.

5.3.1.3 CrossReference

The Cross Reference module is used to work side by side with the search module in the Front-End, this module is the one that solves the referencing issues, it will be able to instantiate a nickname to a person so that all the documents corresponding to the search key would come out. Other than a person's name, it could also instantiate organizations. An example would be when a user types in "Mar Roxas" and clicked "Search" then the system would not only search for "Mar Roxas" but also for his real name which is "Manuel Roxas". Another example would be for the organization, when the user types in "BRC" then the system would know that the user searches for "Blue Ribbon Committee" and it will search in the database for both "BRC" and "Blue Ribbon Committee". The cross reference module uses the parenthesis as boundaries for the marker where shortforms are found. However, in the document, there can be occurrences where in there is a sentence enclosed in parenthesis. In that case, since this module comes after the sentence splitter, this module already knows which belongs to the same sentence and so it would no longer consider that issue as something possible upon extraction. The module also handles names by using quotation marks and finding the set of names in certain locations in certain documents. The cross reference module first finds open parenthesis and once they are found, it would get the short form enclosed in the parenthesis. It then finds the supposedly length of the long form, a formula which was made by Park and Byrd was used. Using the result, the lesser value is selected and it is used as basis in selecting the candidate long form. The short form and long form is then passed to the function made by Schwartz and Hearst as seen in table listing 1. This process is to be repeated for all items in the list of tokenized sentences.

Table 5.5: Statements to find the boundaries of short form

Condition	Action
if(sentenceToken==())	openParenthesis=index of token;
else if(sentenceToken==))	closeParenthesis=index of token;
if (closeParenthesis!=-1 oopenParenthesis!=-1)	refer to Table 5.6

Table 5.6: Statements to find the list of short forms and long forms.

Condition	Action
if(shortForm.size) 1)	for(all the letters in the short- Form) if(letter on shortform is of upper case) shortFormCharLength++;
else	shortFormCharLength = short- Form[0].length;
if(shortForm.size+5 <= short- Form.size*2)	lengthOfLongForm = short- Form.size+5;
else if(shortForm.size*2 < short- Form.size+5)	lengthOfLongForm = short- Form.size*2;

The findBestLongForm function made by Schwartz and Hearst uses the characters in the short form and matches the characters with the long form. Once the character has matched the long form the index parsing through the short form is decremented. The index starts at the end of the short form and long form then backtracks.

```

1 function findBestLongForm(String shortForm, String longForm) {
2     int sIndex;
3     sIndex = shortForm.length() - 1; lIndex = longForm.length() - 1; for ( ; sIndex >=
4         0; sIndex--) {
5         currChar = Character.toLowerCase(shortForm.charAt(sIndex));
6         if (!Character.isLetterOrDigit(currChar)) continue;
7         while ( ((lIndex >= 0) && (Character.toLowerCase(longForm.charAt(lIndex)) !=
8             currChar)) ||
9             ((sIndex == 0) && (lIndex > 0) && (Character.isLetterOrDigit(longForm.charAt(
10                 lIndex - 1)))))
11             lIndex--;
12         return null;
13     }
14     lIndex = longForm.lastIndexOf(" ", lIndex) + 1;
15     return longForm.substring(lIndex);
16 }

```

Listing 1: Find Best Long Form Function of Cross Reference

The names in the documents are found in Notice of Hearing, Hearing Highlights and Agenda. These are pattern based and for each kind document, there lies a different pattern. The pattern for stating names is uniform in the three documents. These are in all caps. The conditions considered with its corresponding action in Table 5.7 gets the formal and informal name of the

Person. The informal name is obtained by getting the tokens between the quotation marks and the formal names are obtained by getting the tokens starting from the first index to the starting quotation mark and followed by the tokens after the quotation mark. If it is a Notice of Hearing document, it looks for "For:" as its marker and starts looking for names there. It stops when it reaches "Please be informed that". It is possible that there are two occurrences of a name in a TokenValue, the mark of the token "HON" is used as a basis on how many times the process will be reiterated per sentence. The Agenda document on the other hand looks for the TokenValue containing "Witnesses/Resource persons" or "Resource Person". This document involves finding the position of the person as well as the organization. The organization and position goes hand in hand in this document. In the Token Value containing the name, the process of finding the formal name and common name of the person is shown in Table 5.8. It is possible in this document that the informal name is not present, however if either, the position or the department is present then this would still be stored in the database. Sometimes the position is indicated by a presence of a dash before it followed by the position the comma and finally the department. However sometimes it is located at the TokenValue following the name in all caps. The module checks if a dash is present in the same TokenValue as the formal name and it gets the position and organization from there. If the sentence following that contains a comma it is considered as the person's department and position. The position comes first so it is the tokens at the first index up until the comma token has been encountered. The department comes after the comma up until the last token. The Hearing highlights follows a different process. It uses SentenceValues instead of token values and the conditions and corresponding actions can be seen on Table 5.9. It splits the sentence where there is a semicolon in between and further split to commas. After splitting the string where commas are present, it checks if there are periods so that there won't be any issues encountered if the comma is due to a mark for a title such as ", Jr." or ", Sr." these are still added to the string that is previously collected, whether it's a position, name or organization. There is always a name however the presence of the position is not always the case. Sometimes there is the department and the rules or conditions to know these things are seen in Table 5.10. This process is reiterated for all the sentences.

Table 5.7: Conditions for getting the Formal and Informal names of the person

Condition	Action
if(token==" ' ' ") quotes==true)	quotes=false;
else if(token==" ' ' ") quotes==false)	quotes=true;
if(begin==true (tokenparagraphNumber == 1 tokenparagraphNumber == 2))	if(quotes==false) FormalName += token; else if(quotes == true) informalName += token;

Table 5.8: Conditions and corresponding actions in gathering of positions and departments

Condition	Action
if(sentence.getTokens().get(i). getToken().equals(","))	commact++;
Continued on next page	

Table 5.8 – continued from previous page

Condition	Action
if(commact==0)	position+=token;
else if(commact\ 0)	department+=token;

Table 5.9: Conditions and corresponding actions in gathering the names, positions and departments in the Hearing highlights document

Condition	Action
if(split string contains ".")	temparr=f[index].split("."); if(temparr.length==1) temptext2=temptext; temptext=""; temptext=f[index] + temptext2; cont=false;
if(cont)	refer to Table 5.10
if(department!=null)	add person to database

Table 5.10: The heuristics for determining if the string is a department, position or a common name in the Hearing Highlights document

Condition	Action
if(index!=0 && per.getDepartment()==null)	department=f[index] + temptext;
else if(index!=0 per.getDepartment()!=null per.getPosition()==null)	position=f[index]
else if(index==0)	commonName=f[index] + temptext formalName=f[index] +temptext

Table 5.11: CoReference Improvement

Condition	Action
if(pronoun is either he, she, her, his, him, hers)	for(all tokens in sentence where pronoun belongs) if(token is all caps) candidate+=token+" ";
if(candidate is more than one word)	winning candidate=candidate;

The input of this module is the list of tokenized sentences. This module goes through each element in the array of tokenized sentences and if it sees an open parenthesis and closing parenthesis, it captures the possible short form enclosed in it. The module would know that it is a correct short form if it has at least two capitalized letters, and it's not a sentence. It's no

longer considered a valid short form if no long form can be found in the same sentence, this is a concept taken from the algorithm of Schwartz and Hearst. Based on the patterns in the document, it is possible that it's not a short-form because of the fact that it's a sentence or it's just a translation of a word.

5.3.1.4 POS Tagger

The POS tagger is the module responsible for assigning part-of-speech tags to the words found in the document. After going through the modules above, the document is currently in tokens and this is necessary for the POS tagger since it needs to read words token for token to assign the corresponding part-of-speech. The module used LingPipe for the POS tagger.

The module made use of an existing corpus named Brown corpus which has files with tagged words. The module then used every file as a training file to create a Hidden Markov Model file. This model is the one responsible for the tagging of part-of-speech to the tokens.

```

function createHMMmodel {
2   initialize LINGPIPE's HmmCharLmEstimator
   declare LINGPIPE's Parser<TagHandler>
4   set the handler of Parser<TagHandler> to be a HmmCharLmEstimator
   open and read the files in the corpus folder for training
6   parse the files using the Parser<TagHandler>
   //create HMM model using the trained HmmCharLmEstimator
8   HmmCharLmEstimator.compile to a .hmm file
}
10
function POSTagger {
12  declare variable hmm using LINGPIPE's HiddenMarkovModel
   read the HMMmodel file
14  cast the input stream into HiddenMarkovModel
   hmm = input stream
16  use LINGPIPE's HMMdecoder to decode the hmm
   decode.tag('any tokens')
18 }

```

Listing 2: POS Tagger Functions

Issues were also encountered during the development of this module. One notable issue is that some tokens are not being tagged correctly. The reason is that for all words or tokens in all caps, the Tagger automatically tags it as proper nouns which should not be the case. Also, another issue is that the implementation of the LingPipe POS Tagger is quite different from the implementation of the preceding modules and as a result some modifications were done.

5.3.1.5 Unknown Word

The unknown word module was built using the algorithm of ANNIE found in its POS tagger. The algorithm of ANNIE exploits the pattern of words and tries to examine the given word's construct. It makes use of the presence of defined suffixes, the presence of 's', capitalization, presence of numerics and '-' in words. words with either '-ed', '-ical', '-ful', '-ary', '-ive', '-ble', '-ic', or '-us' indicate that the word is an adjective. The presence of the dash symbol '-' also signifies that the word that is being examined is an adjective. If the word begins with a capital letter, then it must be a proper noun, this rule however has been added by the research. If the

word ends with a letter 's' then it must be a proper noun. If the word contains numerics then it must be a number. If all these rules fail to satisfy the given word, then the word shall be tagged as a noun.

In the process of researching on the unknown word beforehand, it was mentioned that sometimes the marker “[sic]” occurs in documents therefore more complicated algorithms were included in the plan for the unknown word module. However, according to further research and insights from the Blue Ribbon Committee resource persons, it can only be sighted in dialogues which will not be processed by the system as mentioned in the scope and limitations of the research. the research decided to proceed with the development of ANNIE and not to further implement the other algorithms such as the modified 3-way-method of Elmi and Evans as well as the Tribayes algorithm of Golding and Schabes since they won't be used by the system anyway.

Table 5.12: Unknown Word Pseudocode

Condition	Action
if(word == “.” or “?” or “!” or “,”)	postag of word=“.”;
else if(word=“-”)	postag of word=“-”
if((word's length==1 and initial letter of word is numeric and initial letter of word is NOT an alphabet) word is special character)	postag of word=word;
if(word ends with “ed” or “less” or “ical” or “ful” or “ary” or “est” or “ive” or “ble” or “ic” or “us” or contains “-”)	postag=adjective;
else if(uWord.endsWith(“s”) and first letter is alphabet)	postag=noun;
else	for(int iNumber=0; iNumber<9; iNumber++) (uWord.contains(iNumber+””)) return “cd”;

5.3.1.6 Named Entity Recognition

This module basically associates a token with its corresponding entity. Entities could be time, place, or even person. This done by using LingPipe's Dictionary Mapping named entity recognition. By having a database of entities which serves as a look-up table, the module is able to match a certain token or tokens to its matching entity using Dictionary Mapping.

The look-up table of this module is populated by getting information from different websites found in the internet. Also, the Blue Ribbon Committee provided some list of entities which proved helpful in the construction of the look-up table. Moreover, the Cross-reference module also helps in populating the look-up table by sharing some of the entities it found to the database of the NER.

```

function NERloadEntityDirectories {
2   create String variable entity
   create ArrayList<String> for the directories
4   create ArrayList<String> for the entity types
   open and read file containing entity types
6

```

```

8   while(entity takes the value of file readline) not equal to null) {
    add to directories the value entity+'.txt'
    add to entity type the value entity
10  }
12 }
14 function createNERdictionary{
    initialize LINGPIPE's MapDictionary<String> structure
    for(the size of ArrayList<String> directories) {
16     open and read file with the filename directories[i]
        while((String variables takes the value of file readline) not equal to null) {
18         add to MapDictionary<String> structure value s and entity type[i]
        }
20     }
22 }
24 function NERtagger() {
    initialize LINGPIPE'S ExactDictionaryChunker tagger = new ExactDictionaryChunker(
        createNERdictionary())
    use the existing function of the ExactDictionaryChunker tagger.chunk('any text')
26 }

```

Listing 3: Named Entity Recognition Functions

During the development of the module there was a major shift in the type of named entity recognizer that is supposed to be made. Initially, the named entity recognizer for the system is supposed to be a statistical one, but with the stakeholders involved, the research should also take in to account their needs. The fact that the Blue Ribbon Committee should also be able to add entities made the research realize that the module should be developed in a way that it will be easy for the Blue Ribbon Committee to add entities. As a result, the Dictionary Mapping type of Named Entity Recognizer was chosen.

5.3.2 Parser

The module basically uses the tagged output of the upper modules, more importantly, the POS Tagger, to be able to detect noun phrases and verb phrases. In the case of the system, noun phrases are the target and the module is particularly for detecting noun phrases. Definitely, having the POS Tagger tag tokens is mandatory for the module to work because it relies heavily on the part-of-speech of a certain token to be able to detect. the research used LingPipe and its libraries to come up with a Parser.

One issue on this module is that it relies entirely on the POS Tagger. If the results of the POS Tagger were inaccurate, the output of the Parser will also be inaccurate.

5.3.3 Semantic Tagger

The Semantic Tagger is the module responsible for extracting candidate values for each field found in a certain template of a document. The Semantic Tagger makes use of the outputs provided by the previous modules to determine the correct candidate values. It basically goes over a certain document and finds the candidates by the use of semantic rules or patterns.

Fields of each template can be categorically classified by the complexity in which the information can be extracted from the document. There are fields that are easy to fill since the values that is needed is already part of a template that is used by the Blue Ribbon Committee and therefore, don't require any preprocessing to obtain their respective values. Examples of the easy fields are the hearing number in the agenda document, subject in the hearing highlights document, and referral in the order document. Another set of fields are the slightly harder ones to fill where the values can vary depending on the situation or circumstance presented in the document. However, since the Blue Ribbon Committee follows a certain template for each document, minimal preprocessing is needed for these set of fields.

Table 5.13: Agenda and Scenario Pseudocode

Template Field Type	Condition	Action
Hearing Number	If end of line is HEARING	Hearing Number = first token
Hearing Type	If line contains JOINT PUBLIC	Hearing Type = second to third tokens
	Else	Hearing Type = second token
Hearing Date	If line contains any day of the week and any month	Hearing date = third token to fifth token
Hearing Time	If line contains time : and a or A and m or M	Hearing time = seventh to the last token to last token
Hearing Place	If line has Room or Rooms or Bldg or Floor or City	Loop until line does not contain '.' and gets index of first '.' then Hearing Place = first token to nth token.
Referral	If line contains consideration and referral	Loop while line does not contain WITNESS or RESOURCE PERSON then Referral[] = each line that contains P.S. or Privilege Speech

Table 5.14: Committee Report Pseudocode

Template Field Type	Condition	Action
Committee Report No.	If line starts with COMMITTEE REPORT NO.	number = last token
Committee Report Title	If line starts with SHORT TITLE:	Title = fourth to last token
Committee Report Date	If line starts with Filed On:	Date = fourth to last token
Committee Report Reporting Committees	If line contains COMMITTEE(s) which reported out the bill(s)/resolution(s):	Loop while line does not contain BILL(s) and/or RESOLUTION(s) reported out and taken into consideration committee = each line
Continued on next page		

Table 5.14 – continued from previous page

Template Field Type	Condition	Action
Committee Report Signatures	If line contains MEMBERS Who Failed to Sign the Report; and Reasons:	Signatures = all tokens.

Table 5.15: Hearing Highlight Pseudocode

Template Field Type	Condition	Action
Receiver	If line starts with FOR:	Receiver = all tokens from index of :
Sender	If line starts with FROM:	Sender = all tokens from index of :
Subject	If line starts with SUBJECT:	Subject = all tokens from index of :
Date	If line starts with DATE:	Date = all tokens from index of :
Call to Order	If lines starts with Call to Order:	Call to order = all tokens from index of : up to Adjournment or Suspension. Adjournment = all tokens from Adjournment:

Table 5.16: Hearing Invitation Pseudocode

Template Field Type	Condition	Action
Issue Date	If line contains any month	Issue Date = all tokens.
Referral	If lines contains been referred	Loop until line does not contain P.S. or Privilege Speech referral = each line that starts with P.S. or Privilege Speech
Hearing Date	If token[j] day of the week	Hearing date = token[j+1] to token[j+4]
Hearing Time and Hearing Place	If hearing date not null	Loop while time pattern not found hearing time = all tokens from index of time pattern to index of nearest 'at' hearing place = all tokens from index of nearest 'at' + 1
Sender	If line contains Very truly yours,	Sender = all tokens of next line.

Table 5.17: Notice of Hearing Pseudocode

Template Field Type	Condition	Action
Issue Date	If line contains any month	Issue date = all tokens of the line
Hearing Number	If line starts with NO-TICE	Hearing number = third token
Hearing Type	If line contains JOINT PUBLIC	Hearing type = JOINT PUBLIC
	else	Hearing type = PUBLIC
Referral	If lines contains been referred	Loop until line does not contain P.S. or Privilege Speech referral = each line that starts with P.S. or Privilege Speech
Hearing Date	If line contains request your presence	j = index of a day/time modifier Hearing date = tokens from j+1 to j+4
Hearing Time and Hearing Place	If hearing date not null	j = index of a time pattern k = index of nearest at hearing time = tokens from j to k-1 hearing place = all tokens from k+1

Table 5.18: Order Pseudocode

Template Field Type	Condition	Action
Referral	If line starts with RE:	Referral = all tokens from index of :
Reason	If line contains ORDER	j = find index of {name} is hereby cited in contempt. Reason = first token to jth token proceed to 2 lines before
Hearing Date	If line contains ORDER	Loop while there are valid dates hearing date[] = tokens from index of a day/time modifier to last token
Name		Find identified entities that match the pattern {name} is hereby cited in contempt Name = {name}
Number of Days	If line contains Office of the Senate Sergeant-At-Arms	Number of days = tokens after Office of the Senate Sergeant-At-Arms
Duration	If line contains The Sergeant-At-Arms	If line contains make a return hereof duration = tokens after make a return hereof
Continued on next page		

Table 5.18 – continued from previous page

Template Field Type	Condition	Action
Issue Date and Issue Place	If line contains SO ORDERED.	If line starts with Issued this j = index of this k = index of valid year issue date = tokens from j+1 to k venue = all tokens starting from k+1

Table 5.19: Subpoena Pseudocode

Template Field Type	Condition	Action
Referral	If line starts with IN THE MATTER OF:	Loop while there are valid referrals start = index of P.S. or Privilege Speech end = index of next P.S. or Privilege Speech referral[] = tokens from start to end
Type of Subpoena	If line starts with SUBPOENA DUCES TECUM/AD TESTIFICANDUM	Type = SUBPOENA DUCES TECUM/AD TESTIFICANDUM
	Else if line starts with SUBPOENA AD TESTIFICANDUM	Type = SUBPOENA AD TESTIFICANDUM
	Else if line starts with SUBPOENA DUCES TECUM	Type = SUBPOENA DUCES TECUM
Receiver	If line starts with TO:	Receiver = third token to last token
Receiver's Position		Position = all tokens of line after receiver
Hearing Date	If type = Ad Testificandum or Duces Tecum/Ad Testificandum	i = find line that starts with Day, Date & Time j = index of day/time modifier + 2 Hearing Date = tokens starting from j to index of year.
Hearing Time	If hearing date not null	Hearing Time = all tokens of line after hearing date
Hearing Place	If hearing date not null	Loop while line starts Place: is not found then i++ Hearing Place = tokens after Place: to a jcity entity
Continued on next page		

Table 5.19 – continued from previous page

Template Field Type	Condition	Action
Requested Documents	If type Duces Tecum or Duces Tecum/Ad Testificandum	j=find last index of a jcity entity Loop while line does not contain FAIL NOT UNDER PENALTY OF LAW if first document document[] = all tokens after j else document[] = all tokens

5.3.4 CoReference Resolution

The CoReference Resolution module uses the algorithm of Ruslan Mitkov in order to find what pronouns represent, such problem is coined as anaphora resolution. The input of this module is the output of the Semantic Tagger which could be a whole chunk of sentence or a list of tokens. If this module accepts a chunk of sentence it would have to do an extra step which is to find the location of the sentence in the document and get the corresponding list of tokens with their complete attributes critical to the CoReference resolution module such as the part-of-speech tags and the output of the Preparser which is the chunks of phrases in the sentence. It only processes the item in the array of candidates for templates of that document if a pronoun is found in the given sentence. The module then traces back for the sentences in boundary with the distance defined by Ruslan Mitkov. The sentences considered are those within 3 sentence distance before the sentence wherein the pronoun was found as well as the said sentence itself. All the noun phrases and nouns within the 3-sentence distance are considered candidates while in the sentence containing the pronoun, only the nouns and nounphrases before the pronoun will be considered. There is also the concept of scoring which would determine if the candidate is a winning candidate or not. The winning candidate means that it is the noun or noun phrase that the pronoun is referring to. A winning candidate is the candidate with the highest score if they all tie up then the winning candidate would be determined by additional rules based on distance or usage defined by Ruslan Mitkov just for this kind of case.

Ruslan Mitkov's first rule considers definiteness this rule checks if the candidate is modified by a definite article or by demonstrative or possessive pronouns, as seen in the pseudocode below, the system checks if it the head noun of the noun phrase is preceded by either a demonstrative or possessive pronoun and if it is, it scores 0. The definite article is the article the (Jacobs, 1993). Demonstrative pronouns such as: this, these that and those are pronouns which are used in pointing something or someone (Aarsvold, 1996). Possessive pronouns are the pronouns: mine, ours, yours, your, my, our, his, her, his, hers, its, ones, their, theirs (Jacobs, 1993). Meanwhile indefinite noun phrases are noun phrases modified by an indefinite article, a or an (De Haan, 1989) are penalized by -1. The second rule as prescribed by Ruslan Mitkov uses simple heuristics to determine givenness, which is to score the first noun phrase in a non imperative sentence 1 otherwise the candidate scores nothing. Imperative sentences are determined by definition imperative sentences give you a command or makes an entreaty or request wherein the subject is always you and the only thing that makes the sentence different from most is that the subject is not expressed (Emery, 1961) In the code, imperatives are determined by looking at the first token of the sentence if it is an adverb or a verb. It also looks if it is an adverb because it may come before a verb still making it an imperative sentence. The third rule simply states that if one of the verbs stated by Ruslan Mitkovs list of indicating verbs appeared, the noun phrase that comes after it scores a 1 Verb_set = {discuss, present, illustrate,

identify, summarize, examine, describe, define, show, check, develop, review, re- port, outline, consider, investigate, explore, assess, analyze, synthesize, study, survey, deal, cover}. The fourth rule sees if the phrase is reiterated in the paragraph, if it is reiterated once then it scores 1, if it is reiterated twice or more then it scores 2. The sixth rule states that a non-prepositional noun phrase scores 0 otherwise it is penalized by -1. A prepositional noun phrase are phrases wherein a preposition must be followed by a noun phrase (Stevenson, 1987). In the code the system checks if the noun phrase is preceded by a preposition and if it is, it would be penalized by -1. Rule 7 states that the noun phrase should be in the pattern noun phrase (pronoun), verb and verb, noun phrase (pronoun) such as in Press the key down and turn the volume up.. Press it again wherein the key is the candidate noun phrase and it is the pronoun examined. Rule 8 again follows a pattern prescribed by Ruslan Mitkov called immediate reference, .. (You) Verb1 noun phrase... conjunction (you) verb2 it (conjunction(you) Verb2 it) the ones enclosed in parentheses are optional. In the code, the system finds the necessary components mentioned in the pattern and sees if it matches the pattern given. Rule 9 it is needed to determine what type of sentence is examined whether they be either complex or simple. Complex sentences by definition are sentences which have more than one clause (Radford, 2004). for simplicity and as it was not specified by Ruslan Mitkov, compound-complex sentences which have two or more independent clauses and at least one dependent clause are considered under complex sentences. Simple sentences by definition are sentences which has only one set of subject and predicate. because the algorithm of Ruslan Mitkov only considers complex and simple sentences compound sentences are not strictly counted. Clauses in rule 9 are computed just by counting. To determine the number of clauses in a sentence, the module counts the sets of NP and VP it finds and it looks for subordinators. There are three types of clauses, independent and embedded and subordinate which are under dependent clauses. Dependent clauses have markers such as the wh- pronouns, when, for, while, where, because, since, as if, as though, so, in order, that, so, as, although, even, though, despite, if ,after, before, since and until (Jacobs, 1993). By definition a clause is a sentence unit containing a subject and a finite verb. An independent clause can stand by itself as a sentence and it contains no conjunction or conjunctive word such as subordinators to make it dependent upon another clause. A dependent clause on the other hand contains an expressed or implied conjunctive word and these clauses may act as nouns, adjectives or adverbs (Emery, 1961). In the code, if the function finds more than one clause then it is considered as a complex sentence. If the sentence is complex, The clause preceding the clause containing the pronoun would earn a score of 2 and the noun phrases the previous sentence would earn 1 for the score. The nouns in 2 sentences back would earn 0 and nouns three sentences further back would be penalised by -1.

Some issues encountered upon implementation was that it was unforeseen that the output of the Semantic Tagger module would vary, specifically speaking that the Semantic Tagger would pass objects not containing the part of speech to the co-reference module but that is a minor issue. The major issue encountered in the implementation of this module is that the accuracy of the module is very dependent on the text input as well as the output of the Parser and the POS Tagger module. An issue of the POS Tagger module which affects this module is the wrongly tagged pronouns, if this is the case then the module will just ignore the input not sensing that there is a pronoun in the sentence. The issue of the Parser module which affects this module is that it separates the title (e.g. Mr., Ms., Dr. and other shortened titles used for people) from the noun phrase where its supposed to be therefore it would be possible to tag just the noun phrase. An example would be given Mr. Smith Jones, Smith Jones would be one noun phrase and Mr. would be another noun phrase. In these cases, if the module would get a title then it would take the noun phrase following it and make it as the winning candidate. The limitation of this module is that it can only use anaphora resolution. It can only deal with personal pronouns (such as he,she,they). For instances wherein cataphora resolution is needed,

the algorithm would use the set of conditions and actions as seen in table

The limitation of this module is that it can only use anaphora resolution. It can only deal with personal pronouns (such as he,she,they). The rules made by Ruslan Mitkov which were adapted by the module are written in pseudocode :

Table 5.20: Ruslan Mitkov Rule 1

Condition	Action
if token preceding candidate == definite article or demonstrative or possessive pronouns	candidates score+=0;
else if token preceding candidate == a or an	candidates score+=0;

Table 5.21: Ruslan Mitkov Rule 2

Condition	Action
if sentenceOfCandi- date.isImperative	candidates score+=1;
sentenceOfCandidate.get(0) == candidate	
else	score+=0;

Table 5.22: Ruslan Mitkov Rule 3

Condition	Action
if specialVerbs.contain(sentenceofCandidate.get(candidate.index-1))	candidates score+=1;

Table 5.23: Ruslan Mitkov Rule 4

Condition	Action
if(candidate is noun phrase)	for(all sentences in paragraph where pronoun is found) for(all phrases) if(candidate == phrase) count++;
if(count-1 > 0)	candidates score+=1;

Table 5.24: Ruslan Mitkov Rule 6

Condition	Action
if(noun phrase candidate con- tains preposition)	candidates score+ =0
else	candidates score+=1;

Table 5.25: Ruslan Mitkov Rule 7

Condition	Action
if(token next to pronoun is verb)	verb=token next to pronoun
if(word to the right of candidate is a verb)	if(candidate next to verb equals to candidate) score+=2; candidates.get(p). setRule(1,2);

Table 5.26: Ruslan Mitkov Rule 8

Condition	Action
if(token is verb)	for(all candidates) if(index of token+1==index of candidate) firstVerb==index;
if(candidate's startIndex-2) 0 firstVerb) -1;)	if(token in currsen with startIndex of candidate-2=="you" token in currsen with startIndex of candidate-1==firstVerb) possibleCandidate = candidate;
if(candidate's startIndex-1) 0 firstVerb) -1;)	if(token in currsen with startIndex of candidate-1=="you" token in currsen with startIndex of candidate-1==firstVerb) possibleCandidate = candidate;
if(index of pronoun-3) =0 firstVerb _i -1) if(pos tag of token with index of pronoun-3 is conjunction token with index of pronoun-2=="you" && token with index of pronoun-1 is verb)	setScore=true;
if(pronoun's index-2) =0) if(pos tag of token with index of pronoun-2 is conjunction && token with index of pronoun-1 is verb)	setScore=true;
if(setScore)	if(candidate == possibleCandidate) score+=2 setRule(0,2)

Table 5.27: Ruslan Mitkov Rule 9

Condition	Action
if(sentence type==complex)	clause.getNounPhrases. score+=2; nounphrases in sentence right before current sentence score+=1; nouns three sentences back score-=1;
else if(sentence type==simple)	nounphrases in sentence right before current sentence score+=1; nounphrases two to three sen- tences further back score-=1;

5.3.5 Template Filler

The input of the Template Filler is the output of the Semantic Tagger which contains the input for the fields in the templates. These candidates would be normalized into a standard form, so that the database would be standardized. The dates, time and names would be the one normalized, the dates would have a format of YYYY-MM-DD which could be seen in Table 5.28. The time would be normalized to 24 hour time which is HH:MM:SS as seen in Table 5.29. The names would be normalized to TITLE first then following the person's FIRST NAME, MIDDLE NAME lastly LAST NAME. After normalizing the entity, the module would then be placing it into the database for further searching as seen in Table 5.27. After normalizing, the DBFiller function would be the one that would place the values inside the database (Refer to Listing 4).

An issue encountered during the creation process would be that there are many date formats being used. So the proponent needed to check all the date formats that are usually used and then from that create a standard format. Another issue would be the time, some people write there time in a 24 hour time manner and some write their time in 12 hour time manner in which it would make the output of the system wrong. So the system would be checking if the value of the hour in the time is greater than 12 and has PM or AM in the end. Then it will be converting it to 24 hour time manner before placing the values in the database.

```

DBFiller(int typeOfDocument, arraylist of candidates){
2  check the type of document
  get the values in the candidates
4  normalize dates
  normalize time
6  normalize names
  place the values of the candidates in their corresponding document field
8  execute the Database query to place the values in the database
}
```

Listing 4: Template Filler: DBFiller

Table 5.28: Template Filler: Normalize Name

Condition	Action
if lower part	normalizedValue = tempString
if not lower part	t convert tempString to character array size = 0 string surname while (size < tempStringChar) while (tempStringCgar not equal to ',') surname plus tempStringChar size++ size++; while (size < tempStringChar) normalizedBalue plus tempStringChar size++

Table 5.29: Template Filler: Normalize Date

Condition	Action
if tempString is found in the formats	normalizedValue will get the formatted values return normalized-Value

Table 5.30: Template Filler: Normalize Time

Condition	Action
if parsed hour == 12 or time frame is AM	normalizedValue plus parsed hour
if parsed hour > 12 or time frame is PM	normalizedValue plus parsed hour + 12
if size < tempTimeChar size	normalizedValue plus temptimeChar

5.4 Evaluation System

The evaluation system would be accepting two excel files as input to check the output of the system. The two excel files are expected results and actual results. After opening the excel files, it will be displayed in the table that could be seen in the screen, but before it is displayed the evaluation system would first normalize the items or entities that needs to be normalized. After normalizing the user needs to input the F-measure alpha which is the weight compared to precision and recall. The output of this evaluation system is percentages of precision, recall and f-measure, this percentages shows the accuracy of the output of the system in comparison with the expected results given by the user.

5.5 Entity Relationship Diagram

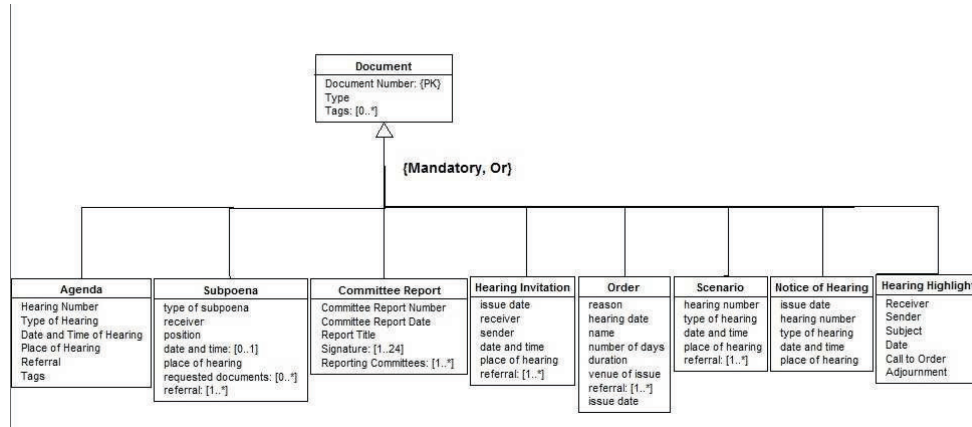


Figure 5.1: ER Diagram

The Information Extraction System gets the input from the Blue Ribbon Committee. The Blue Ribbon Committee gives us a set of documents with types Agenda, Committee Report, Hearing Highlight, Hearing Invitation, Notice of Hearing, Order, Scenario and Subpoena. Each type of document consists of fields that are relevant to them or are important to them. Such as names, addresses, dates, resolutions, reasons, orders, time, and organizations.

5.6 Class Diagram

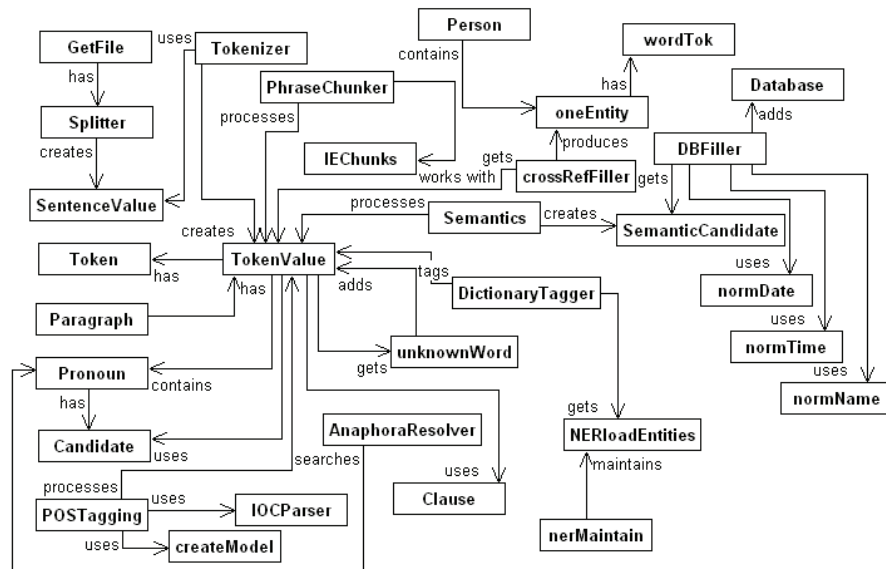


Figure 5.2: Class Diagram

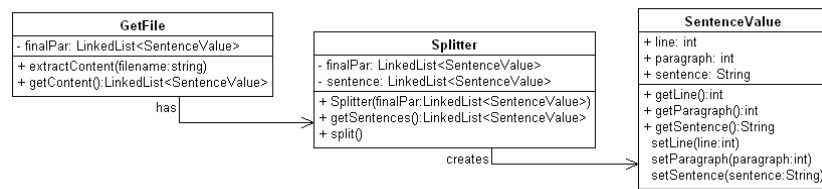


Figure 5.3: Class Diagram: Sentence Splitter

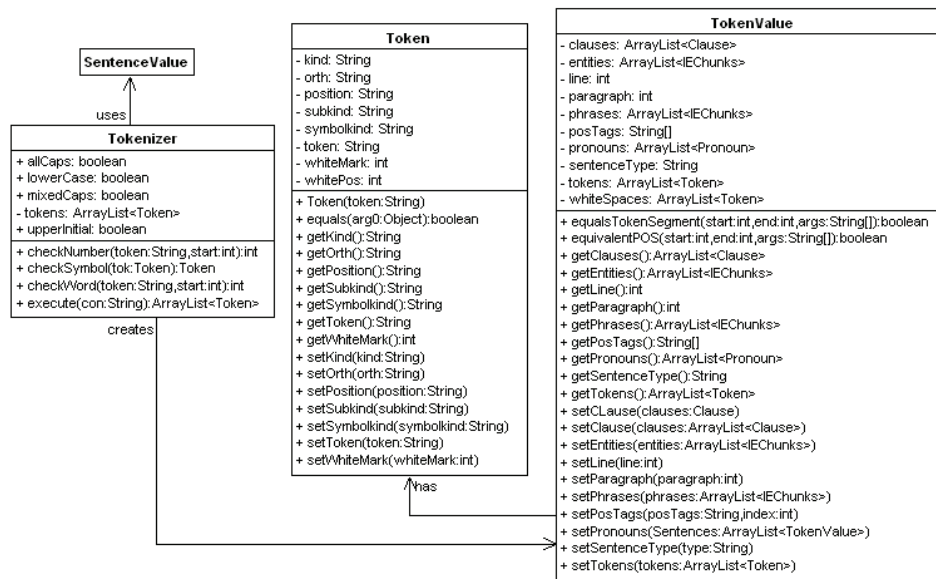


Figure 5.4: Class Diagram: Tokenizer 1

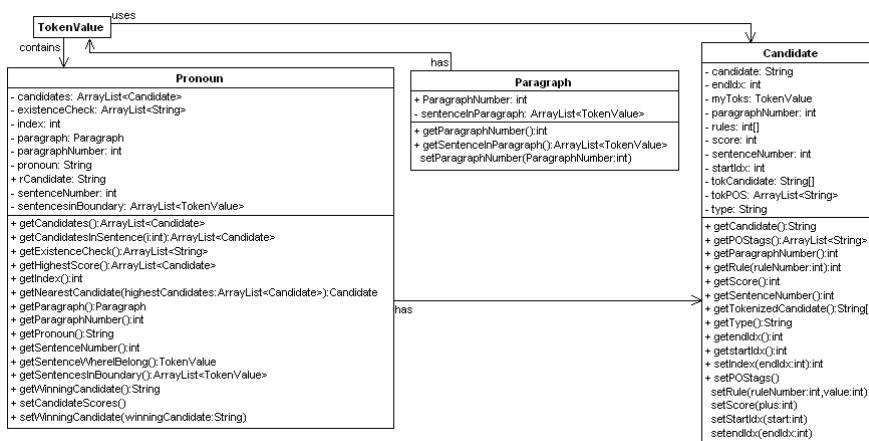


Figure 5.5: Class Diagram: Tokenizer 2

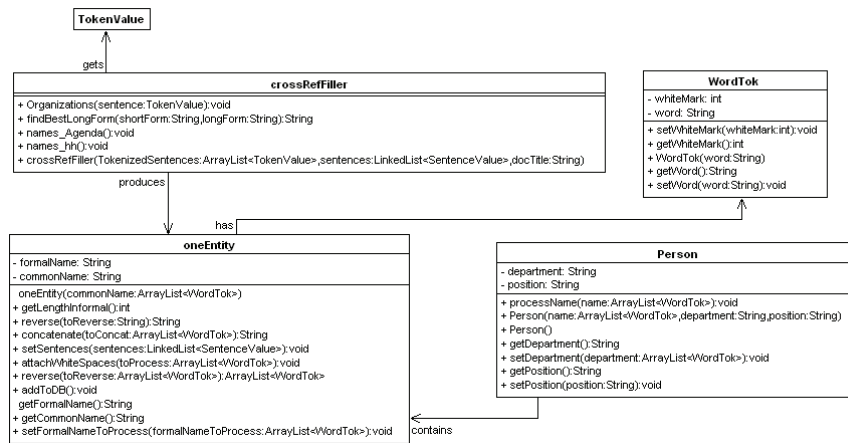


Figure 5.6: Class Diagram: Cross Reference

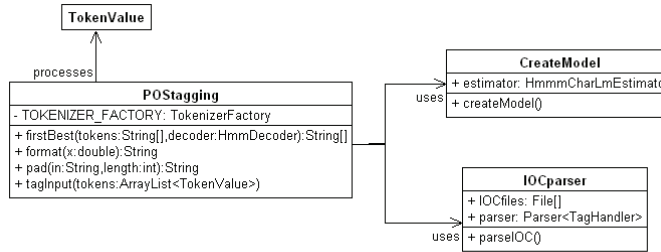


Figure 5.7: Class Diagram: POS Tagger

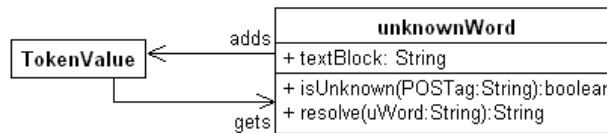


Figure 5.8: Class Diagram: Unknown Word

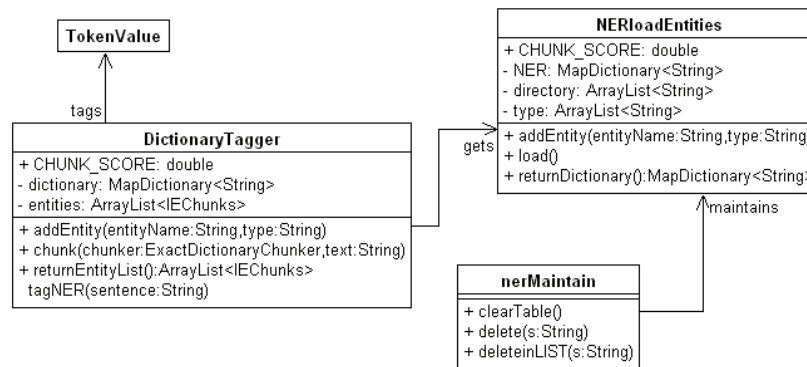


Figure 5.9: Class Diagram: Named Entity Recognition

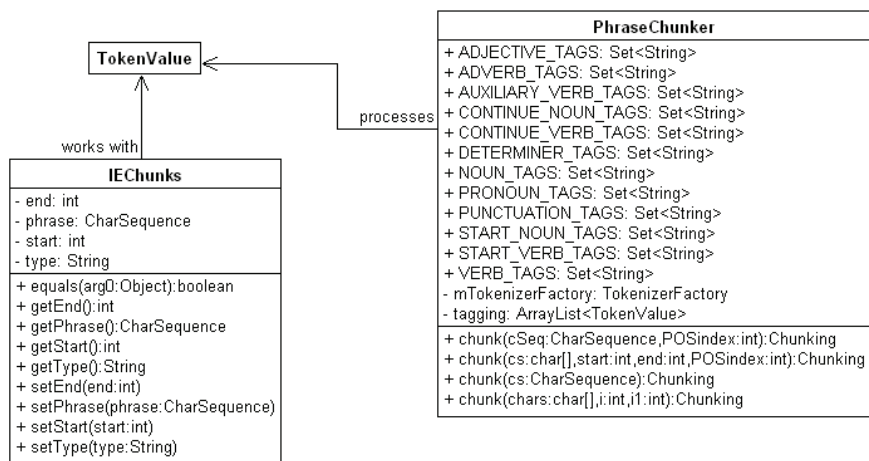


Figure 5.10: Class Diagram: Preparser

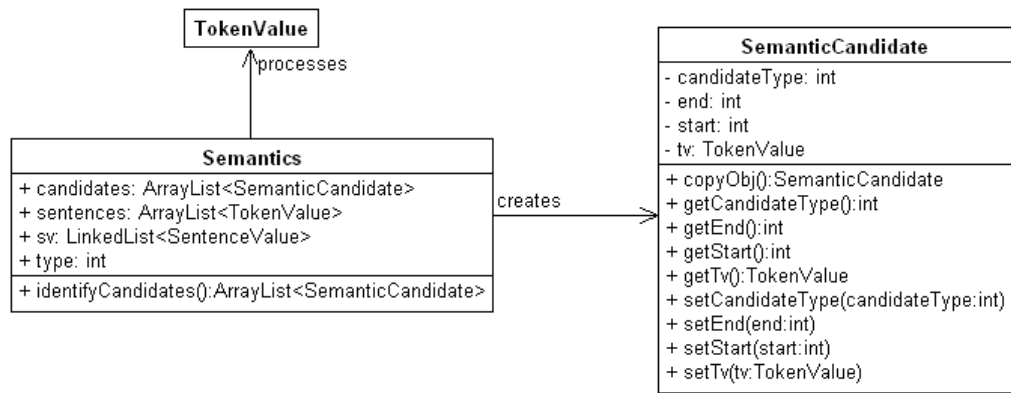


Figure 5.11: Class Diagram: Semantic Tagger

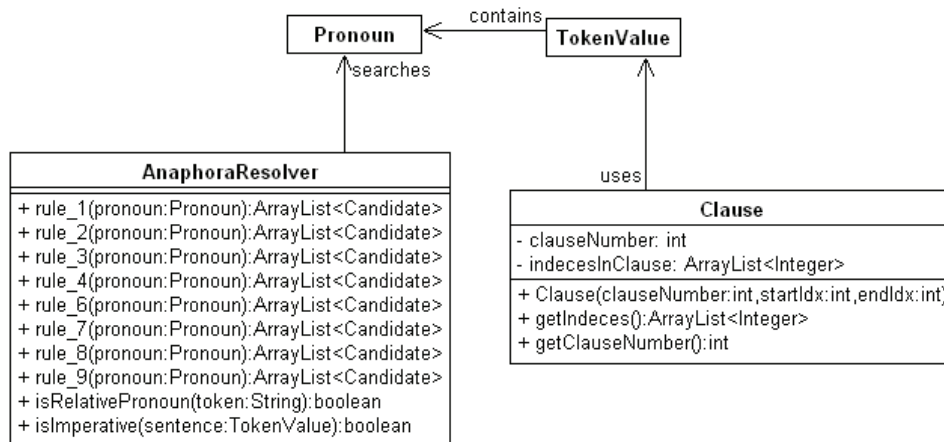


Figure 5.12: Class Diagram: CoReference Resolution

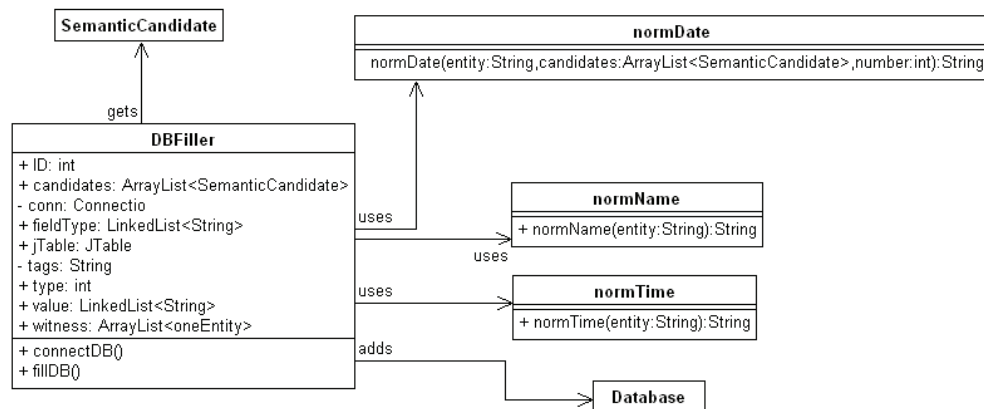


Figure 5.13: Class Diagram: Template Filler

The relationship of the classes in the system could be seen in Figure 5.2. Before everything else, the Sentence Splitter module first gets the file from the directory specified by the user, this document is then divided in to paragraphs and passed to the Splitter. The Splitter class divides the paragraphs into sentences depending on the sentence boundaries. It will only create a sentence whenever it finds a period which indicates that that is the end of the current sentence and the next one would be a whole new sentence. Every sentence created would be added to the Sentence Value class (Refer to Figure 5.3). This class would be indicating the paragraph number and the line number of the current sentence. Next comes the Tokenizer, the Tokenizer will be getting the output of the Sentence Splitter which is the Sentence Value. The Tokenizer would be processing the Sentence Values into tokens. It will be dividing the sentences into tokens. The Tokenizer would be creating Token Values that has Tokens and Pronouns and it creates Candidates. The Tokens indicates the details about the token, such as its kind and its position. For the Pronouns, which indicates whether a certain token is a pronoun or not. The Tokenizer also uses candidates to add values to it for future references in finding the candidate (Refer to Figure 5.4 and Figure 5.5). After, the crossReference module uses the crossRefFiller to produce oneEntity objects containing formalNames and informalNames and afterwards store them in the DataBase to be referred to by the frontEnd module. An object from the oneEntity class may be specifically a person in which case it is possible for the person to have a department or a position which is why the Person class inherits from the oneEntity class. The oneEntity class is composed of objects made using the WordTok class because white spaces are needed to be kept track of (Refer to Figure 5.6). For the POS Tagging module, the system would be getting the Token value that is the output of the Tokenizer, this module would then be creating the model first to be used in the Token Values, the POS Tagger class would be using the parser in order to make the tagging with their own part of speech easier. This module's output would still be the Token Value class with an additional attribute of Part of speech (Refer to Figure 5.7). Unknown Word module, The unknown word module only contains methods that would determine if the given POS tag is unknown or it needs to be resolved further, once determined that it needs to be resolved, the resolve method is then invoked which returns the appropriate posTag (Refer to Figure 5.8). For the Named Entity Recognition module, the Dictionary Tagger would be getting the list of entities in the nerLoadEntities and then it will tag the tokens with their corresponding entity type. The nerLoadEntities would be maintained by the

nerMaintain, the user would be able to add or delete an entity using the nerMaintain (Refer to Figure 5.9). Next, the Preparser has a PhraseChunker class, this class would be chunking the tokens that are to be together in the document such as noun phrases and verb phrases. After finding out the phrases from the Token Value class, there is an IE chunk class which would save the phrases. This class has the phrases and it also indicate on what is the start and end token of the phrase. The output would still be the Token Value class that already includes the phrases formed in this module (Refer to Figure 5.10). Next in line the Semantic Tagger, this module has a Semantic Class which has a function called identifyCandidates, this function finds out the candidates in the document that will be entered in the database. The pseudocode indicates on how the information are being extracted from a specific document. The rules being used are based on the pattern found in the document. Each document has their own specific pattern to follow based on what the client indicated (Refer to Figure 5.11). As for the Coreference module, in each paragraph there are sentences and it also has a paragraph number. For each sentence, there is a possibility that there is a pronoun, if there are pronouns, pronoun objects are made from the Pronoun class. Each pronoun has candidate objects made from the Candidate class. Each candidate has a score as and three special scores from three rules to remember because it is needed if there are candidates that tied up once the scoring has been finalized. These candidates can be first found in the TokenValue objects. Pronouns are passed to the methods contained in the AnaphoraResolver Class. The AnaphoraResolver class contains all the rules of the algorithm of Ruslan Mitkov. It also contains some methods that are necessary to proceed with the scoring with respect to what is prescribed by the algorithm of Ruslan Mitkov such as the isRelativePronoun method and isImperative method. Also, due to the rules of Ruslan Mitkov, clauses in sentences must also be kept track of and these clauses are found in tokenValues (Refer to Figure 5.12). Lastly, Template Filler module has 4 classes which are, DBFiller which fills in the data into the database, normTime which normalizes the time in standard format, normDate which normalizes the date into standard form and normName which normalizes the name into standard format. The input values are from the semantic tagger and then it is first normalized before entered into the database. The way that the system adds into the database is also per document basis. There are certain rules for each document that should be followed before entering into the database (Refer to Figure 5.13).

6 Results and Observations

This section talks about the observations and the outputs of the back-end which performs information extraction and the front-end which is for searching. The research asked for documents in the Blue Ribbon Committee and could only get 100 documents for testing. The evaluation module of the system compares the output to the answer keys provided by the Blue Ribbon Committee. Evaluation is also done on the front-end by means of survey sheets handed down to 30 different respondents who volunteered to test the Front-End.

6.1 Back-End Evaluation

6.1.1 Testing Setup

The research used 50 documents in evaluation. The system supports seven kinds of documents and of those 50 documents, 5 are notice of hearing documents, 5 are agenda documents, 5 are order documents, 12 are subpoena documents, 10 are scenario documents, 8 are hearing invitation documents, and 5 are hearing highlights documents. Each type of document has their own set of fields and one of every type of document was used as basis in obtaining the rules for the various fields for the document of its type. Each notice of hearing document has the same format across every notice of hearing documents and this also applies to the other type of documents. The contents of the documents can be found in Appendix B of this document.

Each document type has its own answer key provided by the Blue Ribbon committee. The evaluation module of the system uses this answer key and compares it with the output. The mechanics of how the evaluation system came up with the results is discussed more in Chapter 5.4.

6.1.2 System Performance Analysis

Table 6.1: Training Data Results

	Expected Result	Actual Result
Type of Subpoena	SUBPOENA duces tecum	SUBPOENA duces tecum
Receiver	Admiral WILFREDO D. TAMAYO	Admiral WILFREDO D. TAMAYO
Position	Commandant, Philippine Coast Guard	Commandant, Philippine Coast Guard
Continued on next page		

Table 6.1 – continued from previous page

	Expected Result	Actual Result
Referral/s	P. S. Res. No. 1452 titled, RESOLUTION DIRECTING THE SENATE COMMITTEE ON ACCOUNTABILITY OF PUBLIC OFFICERS AND INVESTIGATIONS TO INVESTIGATE THE PHILIPPINE COAST GUARD, THE DEPARTMENT OF TRANSPORTATION AND COMMUNICATIONS(DOTC), THE MARITIME INDUSTRY AUTHORITY(MARINA) AND OTHER SIMILAR GOVERNMENT AGENCIES WHO ARE MANDATED BY LAW TO REGULATE AND ENFORCE LAWS RELATING TO MARITIME OPERATIONS AND SAFETY FOR POSSIBLE MALFEASANCE AND NONFEASANCE OF THEIR DUTIES RESULTING IN THE WANTON DISREGARD OF SAFETY RULES AND REGULATIONS BY FERRY OPERATORS THEREBY CAUSING AVOIDABLE SEA TRAGEDIES AND LOSS OF LIVES by Senator Richard J. Gordon	P. S. Res. No. 1452 titled, RESOLUTION DIRECTING THE SENATE COMMITTEE ON ACCOUNTABILITY OF PUBLIC OFFICERS AND INVESTIGATIONS TO INVESTIGATE THE PHILIPPINE COAST GUARD, THE DEPARTMENT OF TRANSPORTATION AND COMMUNICATIONS(DOTC), THE MARITIME INDUSTRY AUTHORITY(MARINA) AND OTHER SIMILAR GOVERNMENT AGENCIES WHO ARE MANDATED BY LAW TO REGULATE AND ENFORCE LAWS RELATING TO MARITIME OPERATIONS AND SAFETY FOR POSSIBLE MALFEASANCE AND NONFEASANCE OF THEIR DUTIES RESULTING IN THE WANTON DISREGARD OF SAFETY RULES AND REGULATIONS BY FERRY OPERATORS THEREBY CAUSING AVOIDABLE SEA TRAGEDIES AND LOSS OF LIVES by Senator Richard J. Gordon

As one can see in Table 6.1, the accuracy of the system from the training data given is 100 percent accurate, the reason for this output is because the system is constructed with the training data so it shows that the system is implemented and tested side by side with the training data. The output of the system is based on the pattern that was given to the research by the Blue Ribbon Committee. The resource person indicated that the pattern is being followed for all the documents, so as long as the pattern is being followed by the implementation of the system, then the output would be accurate and what they had expected.

Table 6.2: Actual Data Results

	Expected Result	Actual Result
Type of Subpoena	SUBPOENA duces tecum	SUBPOENA duces tecum
Receiver	Mr. NAPOLEON I. NAZARENO	Mr. NAPOLEON I. NAZARENO
Position		Ramon Cojuangco Building
Continued on next page		

Table 6.2 – continued from previous page

	Expected Result	Actual Result
Referral/s	P.S. Res. No. 637 (THE ENFORCEMENT OF THE CONTEMPT AND ARREST ORDER ISSUED BY THE SENATE AGAINST FORMER DEPARTMENT OF AGRICULTURE UNDERSECRETARY JOCELYN BOLANTE); P.S. Res. No. 702 (THE ALLEGED PHP728 MILLION FERTILIZER FUND SCAM PRIMARILY FOR THE PURPOSE OF GIVING THE FILIPINO PEOPLE, THROUGH THE SENATE COMMITTEE OF THE WHOLE, THE RIGHT TO HEAR THE TESTIMONY OF FORMER DEPARTMENT OF AGRICULTURE UNDERSECRETARY JOCELYN BOLANTE ON THE MATTER UPON HIS DEPORTATION TO THE PHILIPPINES AND AFTER ISSUANCE OF THE PROPER INVITATION AND/OR COMPLIANCE WITH SUCH OTHER LEGAL PROCESSES THAT MAY BE WARRANTED FOR THE SENATE TO ACQUIRE JURISDICTION OVER HIS PERSON); and Privilege Speech of Sen. Jinggoy Ejercito Estrada titled, A CLARION CALL TO CONSCIENCE delivered on November 10, 2008.	P.S. Res. No. 637 (THE ENFORCEMENT OF THE CONTEMPT AND ARREST ORDER ISSUED BY THE SENATE AGAINST FORMER DEPARTMENT OF AGRICULTURE UNDERSECRETARY JOCELYN BOLANTE); P.S. Res. No. 702 (THE ALLEGED PHP728 MILLION FERTILIZER FUND SCAM PRIMARILY FOR THE PURPOSE OF GIVING THE FILIPINO PEOPLE, THROUGH THE SENATE COMMITTEE OF THE WHOLE, THE RIGHT TO HEAR THE TESTIMONY OF FORMER DEPARTMENT OF AGRICULTURE UNDERSECRETARY JOCELYN BOLANTE ON THE MATTER UPON HIS DEPORTATION TO THE PHILIPPINES AND AFTER ISSUANCE OF THE PROPER INVITATION AND/OR COMPLIANCE WITH SUCH OTHER LEGAL PROCESSES THAT MAY BE WARRANTED FOR THE SENATE TO ACQUIRE JURISDICTION OVER HIS PERSON); and Privilege Speech of Sen. Jinggoy Ejercito Estrada titled, A CLARION CALL TO CONSCIENCE delivered on November 10, 2008.

On the other hand, when the actual results were evaluated, there came out one to two fields which did not match with the expected result (refer to Table 6.2). From the type of documents given by the Blue Ribbon Committee, Agenda, Committee Report, Hearing Highlight, Notice of Hearing, Order and Scenario have a perfect score. While for the other two documents being processed by the system which are Hearing Invitation and Subpoena, these are the documents which show some change in their accuracy based on the training data accuracy, the accuracy of the Subpoena is 85% while the accuracy for the Hearing Invitation is 83%. With these percentages the overall accuracy of the system with the actual data would be 95.42%. The reason for a lower rating than the perfect score is because of the values inside the document. There are some instances wherein the personnel would eliminate some values from the document and just write a line or leave it blank. Since the system is pattern-based, the system instead of getting the value "N/A", the system would be getting a line or maybe the next value to be seen that follows the pattern.

Every item in the output of the system is counted with a score of one, which denotes that if there are words added or removed from the desired then the score would automatically be removed. The reason that the research had decided to use this kind of error scoring is because the research thinks that the accuracy of the system would be much higher, since it is an all or nothing basis. Whether the system gets the exact value that the client wants or it does not count a score for the item at all. Compared to the evaluation method used by Legal TrUTHS, they penalize the score if there are more or less words compared to the expected output of the system. With its method, there would always be a score for a given item.

6.1.3 Modular Testing

6.1.3.1 Sentence Splitter

Table 6.3: Sentence Splitter Modular Test

Paragraph	Line	Expected	Actual
This Spot Report is divided into three parts. The first part is with regard to the parallelism of the Swine Scam with the Fertilizer Fund Scam. The second part is the Commission on Audit (COA) Report on Quedancor. Finally, the third part is with regard to the loan process Quedancor.	Line 1	This Spot Report is divided into three parts.	This Spot Report is divided into three parts.
	Line 2	The first part is with regard to the parallelism of the Swine Scam with the Fertilizer Fund Scam.	The first part is with regard to the parallelism of the Swine Scam with the Fertilizer Fund Scam.
	Line 3	The second part is the Commission on Audit (COA) Report on Quedancor. Finally, the third part is with regard to the loan process Quedancor.	The second part is the Commission on Audit (COA) Report on Quedancor. Finally, the third part is with regard to the loan process Quedancor.
Continued on next page			

Table 6.3 – continued from previous page

Paragraph	Line	Expected	Actual
The Sergeant-At-Arms is hereby directed to carry out and implement this Order and make a return hereof within twenty four (24) hours from its enforcement.	Line 1	The Sergeant-At-Arms is hereby directed to carry out and implement this Order and make a return hereof within twenty four (24) hours from its enforcement.	The Sergeant-At-Arms is hereby directed to carry out and implement this Order and make a return hereof within twenty four (24) hours from its enforcement.

The output of this module is similar to what is expected. The expected output is done manually by the proponent and is compared with the output of the module. From Table 6.3, one can see in the first paragraph, when manually divided, the paragraph would be divided into 3 lines or sentences. When compared to the output of the system, the output is the same as the expected result. One when can see that the module divides the paragraph the same as a person divides a paragraph. The same goes with the second paragraph, when manually divided it does not have more than a sentence. Which means that the output should be consisting of one line or sentence and it is the output of the module, this could show the accuracy of the system is 3 out of 3 and 1 out of 1 which equates to a 100% accuracy.

6.1.3.2 Tokenizer

Table 6.4: Tokenizer Modular Test

Line	Expected Token	Actual Token
TUESDAY, 25 NOVEMBER 2008	TUESDAY	TUESDAY
	,	,
	25	25
	NOVEMBER	NOVEMBER
	2008	2008
SENATOR AMBROSIO PADILLA ROOM, 2ND FLOOR, SEN- ATE, GSIS BLDG., PASAY CITY	SENATOR	SENATOR
	AMBROSIO	AMBROSIO
	PADILLA	PADILLA
	ROOM	ROOM
	,	,
Continued on next page		

Table 6.4 – continued from previous page

Line	Expected Token	Actual Token
	2ND	2ND
	FLOOR	FLOOR
	,	,
	SENATE	SENATE
	,	,
	GSIS	GSIS
	BLDG	BLDG
	.	.
	,	,
	PASAY	PASAY
	CITY	CITY

The actual tokens produced by the module are similar to the expected output which is manually done by the proponent. As can be seen in Table 6.4, the first line is tokenized by the system the same way it was manually tokenized. The system tokenized the according to word, number, and symbol. The same can be seen in the next line. When the system tokenized the second line, it still tokenized according to word, number, and symbol. This would show the accuracy of the system which are 5 out of 5 and 16 out of 16 which would translate to a 100% accuracy.

6.1.3.3 Cross Reference

The Cross Reference module does not have any issue other than it can't handle abbreviations which have no pattern in relevance to its formal name (i.e. COMMITTEE ON ACCOUNTABILITY OF PUBLIC OFFICERS AND INVESTIGATIONS(Blue Ribbon)). It has not yet occurred upon testing, but it has already been declared in the paper of Schwartz and Hearst that given Thyroid transcription factor 1 (TTF-1). The algorithm would fail again because thyroid won't be found refer to Listing 1 for the pseudocode of the algorithm of Schwartz and Hearst. Refer to Table 6.5 and Table 6.6 for the analysis of the results of this module.

Table 6.5: Cross Reference Person Modular Test

Data Input	Expected Results	Actual Results
<i>For:</i> MEMBERS HON. MANUEL LITO M. LAPID HON. FRANCIS CHIZ G. ESCUDERO HON. FRANCIS N. PANGILINAN HON. COMPAERA PIA S. CAYETANO HON. ANTONIO SONNY F. TRILLANES IV HON. JOKER P. ARROYO HON. ALAN PETER COMPAERO CAYETANO EX OFFICIO HON. AQUILINO Q. PIMENTEL JR. - Minority Leader HON. JUAN MIGUEL F. ZUBIRI - Majority Leader HON. JINGGOY EJERCITO ESTRADA - President Pro Tempore GUEST SENATORS HON. JUAN PONCE ENRILE - Senate President HON. MANNY VILLAR HON. BENIGNO NOYNOY S. AQUINO	properName: MANUEL LAPID commonName: LITO LAPID properName: FRANCIS ES- CUDERO commonName: CHIZ ESCUD- ERO properName: PIA S. CAYETANO commonName: COMPANERA CAYETANO properName: ANTONIO TRILLANES IV commonName: SONNY TRIL- LANES properName: ALAN PETER CAYETANO commonName: COMPANERO CAYETANO properName: BENIGNO AQUINO commonName: NOYNOY AQUINO	properName: MANUEL LAPID commonName: LITO LAPID properName: FRANCIS ES- CUDERO commonName: CHIZ ESCUD- ERO properName: PIA S. CAYETANO commonName: COMPANERA CAYETANO properName: ANTONIO TRILLANES IV commonName: SONNY TRIL- LANES properName: ALAN PETER CAYETANO commonName: COMPANERO CAYETANO properName: BENIGNO AQUINO commonName: NOYNOY AQUINO

Table 6.6: Cross Reference Organization Modular Test

Data Input	Expected Results	Actual Results
Department of Agriculture(DOA) Commission on Elections(COMELEC) COMMITTEE ON ACCOUNTABILITY OF PUBLIC OFFICERS AND INVESTIGATIONS(Blue Ribbon)	ProperName: Department of Agriculture commonName: DOA properName: Commission on Elections commonName: COMELEC	ProperName: Department of Agriculture commonName: DOA properName: Commission on Elections commonName: COMELEC

6.1.3.4 POS Tagger

Table 6.7: POS Tagger Modular Test

Line	Token	Expect Tag	Actual Tag
Vice President for Physical Resources Office, GSIS	Vice	jj adjective	jj adjective
	President	nn singular noun	nn singular noun
	For	in preposition	in preposition
	Physical	jj adjective	jj adjective
	Resources	nns plural noun	nns plural noun
	Office	nn singular noun	nn singular noun
	,	, - literal comma	, - literal comma
	GSIS	np singular proper noun	nn singular noun
TUESDAY , 25 NOVEMBER 2008	TUESDAY	nr singular adverbial noun	uh - interjection
	,	, - literal comma	, - literal comma
	25	cd cardinal number	cd cardinal number
	NOVEMBER	np proper noun	nps plural proper noun
	2008	cd cardinal number	cd cardinal number

The accuracy of this module lies heavily on the tagged files in the Brown Corpus. The reason why some tokens are not being tagged properly is because the tokens/words are not found in the tagged files in the Brown Corpus. If a word is not found in the tagged files, it is automatically tagged as a noun, which is somehow not a problem because most of the words in the documents of the Blue Ribbon Committee that are not found in the corpus are nouns. As seen in Table 6.7, the first example got 7 out of 8 correct answers for a percentage of 87.5% in accuracy. The module missed the word GSIS for the reason that it is not found in the tagged files

of the Brown Corpus. The system was not able to tag it correctly, but the result is still plausible since GSIS is a proper noun and the rule of the module is that if an unknown word is found, it is tagged a noun and a proper noun is still a noun.

6.1.3.5 Unknown Word

Table 6.8: Unknown Word Modular Test

Data Input	Expected Results	Actual Results
BILL_nil	BILL_nn	BILL_nn
(_nil	(_((_(
S_nil	S_S	S_S
)_nil)_))_)
And_nil	And_nn	And_nn
/_nil	/_	/_

The unknown word module of ANNIE is very straightforward. However, the tagging of ANNIE is very simple compared to the tags produced by LingPipe. Perhaps it would be more accurate if more rules were added in the module to have the same specificity of the tagging of LingPipe. Refer to Table 6.8 for the analysis of this module.

6.1.3.6 Named Entity Recognition

Table 6.9: Named Entity Recognition Modular Test

Line	Expected Tag	Actual Tag
Privilege Speech of Sen. Jinggoy Ejercito Estrada titled A CLARION CALL TO CONSCIENCE delivered 10 November 2008	Sen. Jinggoy Ejercito Estrada<Senator>	Sen. Jinggoy Ejercito Estrada<Senator>
Vice President for Physical Resources Office, GSIS	Physical Resources Office<Office>; GSIS<Organization>	Physical Resources Office<Office>; GSIS<Organization>

This module uses the Exact Dictionary Matcher of LingPipe to tag the noun phrases with their corresponding entities. As said in chapter 5, it is required to enter the entities in the Dictionary for the system to be able to recognize these tokens and tag them correctly. If an entity is not in the database, then the module would not be able to tag it. As seen in Table 6.9, the module was able to accurately tag the phrases with their corresponding entities. Therefore, it is sure to always have 100% accuracy for as long as the expected entities are recorded in the database.

6.1.3.7 Preparser

Table 6.10: Preparser Modular Test

Line	Expected Output	Actual Output
Privilege Speech of Sen. Jinggoy Ejercito Estrada titled A CLARION CALL TO CONSCIENCE delivered 10 November 2008	Noun phrase Privilege Speech	Noun phrase Privilege Speech
	Noun phrase Sen. Jinggoy Ejercito Estrada	Noun phrase Sen. Jinggoy Ejercito Estrada
	Verb phrase titled	Verb phrase titled
	Noun phrase A CLARION CALL TO CONSCIENCE	Noun phrase A CLARION CALL TO CONSCIENCE
	Verb phrase delivered	Verb phrase delivered
	Noun phrase 10 November 2008	Noun phrase 10 November 2008

This module is done by using the predefined methods and classes of LingPipe. Given the sentence in Table 6.10, the module is expected to divide it into 4 noun phrases and 2 verb phrases. As seen in the actual output, the module is successful in dividing it into 4 noun phrases and 2 verb phrases thus with the 100% accuracy. The accuracy of this module is highly dependent on the results of the POS Tagger since it uses its tags to determine the start of a noun phrase and verb phrase.

6.1.3.8 Semantic Tagger

Table 6.11: Semantic Tagger Modular Test

Input	Field Type	Expected	Actual
SECOND PUBLIC HEARING TUESDAY, 25 NOVEMBER 2008 10:00 A.M. SENATOR AMBROSIO PADILLA ROOM, 2ND FLOOR, SENATE, GSIS BLDG., PASAY CITY	Hearing No.	SECOND	SECOND
	Hearing Type	PUBLIC	PUBLIC
	Date of Hearing	25 NOVEMBER 2008	25 NOVEMBER 2008
	Time of Hearing	10:00 A.M.	10:00 A.M.
Continued on next page			

Table 6.11 – continued from previous page

Input	Field Type	Expected	Actual
	Place of Hearing	SENATOR AMBROSIO PADILLA ROOM, 2ND FLOOR, SENATE, GSIS BLDG., PASAY CITY	SENATOR AMBROSIO PADILLA ROOM, 2ND FLOOR, SENATE, GSIS BLDG., PASAY CITY
For failure to appear and testify in the Committee's hearing on Tuesday, September 18, 2007; Thursday, September 20, 2007; Thursday, October 25, 2007 and Tuesday, November 20, 2007, despite personal notice and a Subpoenas Ad Testificandum sent to and received by him, which thereby delays, impedes and obstructs, as it has in fact delayed, impeded and obstructed the inquiry into the subject reported irregularities, AND for failure to explain satisfactorily why he should not be cited for contempt (Neri letter of 29 November 2007, herein attached) ROMULO L. NERI is hereby cited in contempt of this Committee and ordered arrested and detained in the Office of the Senate Sergeant-At-Arms until such time that he will appear and give his testimony.	Reason	For failure to appear and testify in the Committee's hearing on Tuesday, September 18, 2007; Thursday, September 20, 2007; Thursday, October 25, 2007 and Tuesday, November 20, 2007, despite personal notice and a Subpoenas Ad Testificandum sent to and received by him, which thereby delays, impedes and obstructs, as it has in fact delayed, impeded and obstructed the inquiry into the subject reported irregularities, AND for failure to explain satisfactorily why he should not be cited for contempt (Neri letter of 29 November 2007, herein attached)	For failure to appear and testify in the Committee's hearing on Tuesday, September 18, 2007; Thursday, September 20, 2007; Thursday, October 25, 2007 and Tuesday, November 20, 2007, despite personal notice and a Subpoenas Ad Testificandum sent to and received by him, which thereby delays, impedes and obstructs, as it has in fact delayed, impeded and obstructed the inquiry into the subject reported irregularities, AND for failure to explain satisfactorily why he should not be cited for contempt (Neri letter of 29 November 2007, herein attached)
	Name	ROMULO L. NERI	ROMULO L. NERI
Continued on next page			

Table 6.11 – continued from previous page

Input	Field Type	Expected	Actual
	Hearing Dates	September 18, 2007; September 20, 2007; October 25, 2007; November 20, 2007	September 18, 2007; September 20, 2007; October 25, 2007; November 20, 2007
	Number of Days	until such time that he will appear and give his testimony.	until such time that he will appear and give his testimony.

The expected output which is given by the client is compared to the actual output of the system to compute the systems accuracy which could be seen in Table 6.11. In the first example given above which is a part of an agenda document, the system extracted from the document the values that are expected for the fields of Hearing Number, Hearing Type, Date of Hearing, Time of Hearing, and Place of Hearing. As for the second example which is a part of an order document, the system also extracted the values that are expected to fill the fields of the order template. For both examples, the system got 4 out of 4 for each example which is translated to a 100% accuracy.

6.1.3.9 Coreference Resolution

The rules of this module were tested individually and the test script could be seen in Table 6.12. Many aspects affect the accuracy of the CoReference resolution module. Though the module followed Ruslan Mitkov's algorithm, the preprocessing of the entities, i.e. the tagging, phrase chunking and the numbering of paragraphs is not the same, also in the lexical iteration rule, it could be further improved by implementing additional algorithms to consider synonymous phrases. The algorithm also cannot handle cataphoric pronouns, it can only handle anaphora resolution in personal pronouns. Some addons to the algorithm were made to increase the accuracy of the system such as the double checking if the token passed to this module is a personal pronoun or not. Refer to Table 6.12 and Table 6.13 for the analysis of the module.

Table 6.12: Test script for unit testing of Coreference Module with Ruslan Mitkovs rules

Test Description	Data Input	Expected Results	Actual Results
rule 1 : definiteness noun phrases modified by a definite article such as (the) scores 0 while the noun phrases modified by an indefinite article (a or an) scores -1	The Princess has a pet cat. Shelly is what Jaja calls it.	Candidates: Shelly score: 0 Jaja score: 0 Jaja score: 0 Shelly score: 0 Princess score: 0 Pet score: 0 cat score: 0 The Princess score: 0 a pet cat score: -1	Candidates: Shelly score: 0 Jaja score: 0 Jaja score: 0 Shelly score: 0 Princess score: 0 Pet score: 0 cat score: 0 The Princess score: 0 a pet cat score: -1
rule 2: givenness first non-imperative noun phrase in the sentence gets +1 otherwise 0	The key is beautiful. Put the keychain on it.	Candidates: the keychain score: 0 keychain score: 0 key 1 noun 2 score: 0 The key score: 1 beautiful score: 0	Candidates: the keychain score: 0 keychain score: 0 key 1 noun 2 score: 0 The key score: 1 beautiful score: 0
rule 3: indicating verbs noun phrases after an element in a set of specific verbs defined by Ruslan Mitkov such as describe define and show shall get +1 in the score	Wilhansen discussed important matters because they are important.	Candidates: I nounPhrase score: 0 important matters score: 1 matters score: 0	Candidates: I nounPhrase score: 0 important matters score: 1 matters score: 0
Continued on next page			

Table 6.12 – continued from previous page

Test Description	Data Input	Expected Results	Actual Results
rule 4: lexical iteration Candidate noun phrases that have been repeated in the same paragraph will gain 2 points	The jar is blue. The jar is very colorful. Maynard is holding the jar. Wil is as tall as it.	Candidates: Wil 0 1 score: 0 tall 4 score: 0 Wil score: 0 Maynard score: 0 jar score: 0 Maynard score: 0 the jar score: 2 jar score: 0 The jar score: 2 colorful 5 score: 0	Candidates Wil 0 1 score: 0 tall 4 score: 0 Wil score: 0 Maynard score: 0 jar score: 0 Maynard score: 0 the jar score: 2 jar score: 0 The jar score: 2 colorful 5 score: 0
rule 6: pure non-prepositional noun phrases get +0 otherwise -1	Insert the cassette into the VCR making sure it is suitable for the length of recording.	Candidates: the cassette score: 0 the VCR score: -1 VCR score: 0 cassette score: 0	Candidates: the cassette score: 0 the VCR score: -1 VCR score: 0 cassette score: 0
rule 7: collocation pattern preference if the nounphrase belongs to the patterns: NP(pronoun) verb or verb NP(pronoun) nounphrase receives +2	Press the key down and turn the volume up... Press it again.	key score: 0 volume score: 0 the key score: 2 the volume score: 0	key score: 0 volume score: 0 the key score: 2 the volume score: 0
Continued on next page			

Table 6.12 – continued from previous page

Test Description	Data Input	Expected Results	Actual Results
rule 8: immediate reference follows this pattern "...(You) V I NP ... con (you) V2 it (con (you) V3 it)", where con and/or/before/after...,	<p>To print the paper, you can stand the printer up or lay it flat.</p> <p>To turn on the printer, press the Power button and hold it down for a moment.</p> <p>Unwrap the paper, form it and align it, then load it into the drawer.</p>	<p>the paper score: 2</p> <p>you score: 0</p> <p>the printer score: 0</p> <p>printer score: 0</p> <p>paper score: 0</p> <p>the printer score: 0</p> <p>the Power button score: 2</p> <p>button score: 0</p> <p>Power score: 0</p> <p>printer score: 0</p> <p>the paper score: 2</p> <p>paper score: 0</p>	<p>the paper score: 2</p> <p>you score: 0</p> <p>the printer score: 0</p> <p>printer score: 0</p> <p>paper score: 0</p> <p>the printer score: 0</p> <p>the Power button score: 2</p> <p>button score: 0</p> <p>Power score: 0</p> <p>printer score: 0</p> <p>the paper score: 2</p> <p>paper score: 0</p>

Continued on next page

Table 6.12 – continued from previous page

Test Description	Data Input	Expected Results	Actual Results
rule 9: referential distance In complex sentences, noun phrases in the previous clause are the best candidate for the antecedent of an anaphor in the subsequent clause, followed by noun phrases in the previous sentence, then by nouns situated 2 sentences further back and finally nouns 3 sentences further back (2, 1, 0, -1). For anaphors in simple sentences, noun phrases in the previous sentence are the best candidate for antecedent, followed by noun phrases situated 2 sentences further back and finally nouns 3 sentences further back (1, 0, -1).	Jan remembered that day. The day was a Sunday morning. Wil placed the flowers in the vase. The flowers were so beautiful that they almost withered.	The flowers score: 2 beautiful score: 2 flowers score: 0 Wil score: 1 flowers score: 1 vase score: 1 Wil score: 1 the flowers score: 1 the vase score: 1 Sunday score: 0 morning score: 0 It score: 0 a Sunday morning score: 0	The flowers score: 2 beautiful score: 2 flowers score: 0 Wil score: 1 flowers score: 1 vase score: 1 Wil score: 1 the flowers score: 1 the vase score: 1 Sunday score: 0 morning score: 0 It score: 0 a Sunday morning score: 0

Table 6.13: Test script for module test of CoReference module

Data Input	Expected Results	Actual Results
SUBPOENA_nps duces.vb tecum.ppo	No result	No result
*tecum was marked accidentally by the pos tagger as a pronoun		
Continued on next page		

Table 6.13 – continued from previous page

Data Input	Expected Results	Actual Results
RESOLUTION DIRECTING THE SENATE COMMITTEE ON ACCOUNTABILITY OF PUBLIC OFFICERS AND INVESTIGATIONS TO INVESTIGATE THE PHILIPPINE COAST GUARD, THE DEPARTMENT OF TRANSPORTATION AND COMMUNICATIONS (DOTC), THE MARITIME INDUSTRY AUTHORITY (MARINA) AND OTHER SIMILAR GOVERNMENT AGENCIES WHO ARE MANDATED BY LAW TO REGULATE AND ENFORCE LAWS RELATING TO MARITIME OPERATIONS AND SAFETY FOR POSSIBLE MALFEASANCE AND NONFEASANCE OF THEIR DUTIES RESULTING IN THE WANTON DISREGARD OF SAFETY RULES AND REGULATIONS BY FERRY OPERATORS THEREBY CAUSING AVOIDABLE SEA TRAGEDIES AND LOSS OF LIVES by Senator Richard J. Gordon	THEIR = THE PHILIPPINE COAST GUARD, THE DEPARTMENT OF TRANSPORTATION AND COMMUNICATIONS (DOTC), THE MARITIME INDUSTRY AUTHORITY (MARINA) AND OTHER SIMILAR GOVERNMENT AGENCIES	THEIR = THE DEPARTMENT OF TRANSPORTATION AND COMMUNICATIONS
In this connection, may we request your presence during the third public hearing thereon on Thursday, January 7, 2010 at 1:00 pm at the Senator Jose P. Laurel Room, 2nd Floor, Senate of the Philippines, GSIS Bldg., Roxas Blvd., Pasay City	We = blue ribbon	We = this connection
For failure to appear and testify in the Committees hearing on Tuesday, September 18, 2007; Thursday, September 20, 2007; Thursday, October 25, 2007 and Tuesday, November 20, 2007, despite personal notice and a Subpoenas Ad Testificandum sent to and received by him, which thereby delays, impedes and obstructs, as it has in fact delayed, impeded and obstructed the inquiry into the subject reported irregularities, AND for failure to explain satisfactorily why he should not be cited for contempt (Neri letter of 29 November 2007, herein attached) ROMULO L. NERI is hereby cited in contempt of this Committees and ordered arrested and detained in the Office of the Senate Sergeant-At-Arms until such time that he will appear and give his testimony.	Him=Romulo Neri He=Romulo Neri It=failure	Him=ROMULO NERI He=ROMULO NERI It=failure

6.1.3.10 Template Filler

Table 6.14: Test script for unit testing of Coreference Module with Ruslan Mitkovs rules

Input	Field Type	Expected Results	Actual Results
SECOND PUBLIC HEARING TUESDAY, 25 NOVEM- BER 2008 10:00 A.M. SENATOR AMBROSIO PADILLA ROOM, 2ND FLOOR, SENATE, GSIS BLDG., PASAY CITY	Hearing No.	Second	Second
	Hearing Type	Public	Public
	Date and Time of Hearing	2008-11-25 10:00:00	2008-11-25 10:00:00
	Place of Hear- ing	SENATOR AMBROSIO PADILLA ROOM, 2ND FLOOR, SEN- ATE, GSIS BLDG., PASAY CITY	SENATOR AMBROSIO PADILLA ROOM, 2ND FLOOR, SEN- ATE, GSIS BLDG., PASAY CITY

The accuracy of this module is solved by getting a certain part of the document, and get the expected result given by the client. As seen in Table 6.14, the expected result would then be compared with the actual result. They should not have any difference with regards to the words. As one can see, the part of the document could be divided into the Hearing Number, Hearing Type, Date and Time of Hearing and Place of Hearing. The expected result for this part is the same as the actual result so the accuracy of this module is 4 out of 4 a hundred percent.

6.1.4 Actual Data Results

After testing the system with the test data, the research asked for another set of documents that would be used for testing other than the initial test data, the number of additional documents given to the research are all in all 50 documents. As of now, the Blue Ribbon Committee are still giving us new sets of documents for testing. After testing the system with the new test data, the output of the system are mostly the same with only a small part different as could be seen in Table 6.2. The reason behind the difference is that because with the initial data given to us, the research are not able to see those kinds of changes. There are certain parts in the new test data that are not found in our training data and initial test data.

Other fields were also decided to be added to the current templates which made the research tweak a little with the system. There are only a few changes to correspond to the new test

data. So overall, the testing of other documents gives a good number as accuracy because the system is getting the information that is needed by the client or user.

6.1.5 Comparison with Legal TrUTHS Architecture

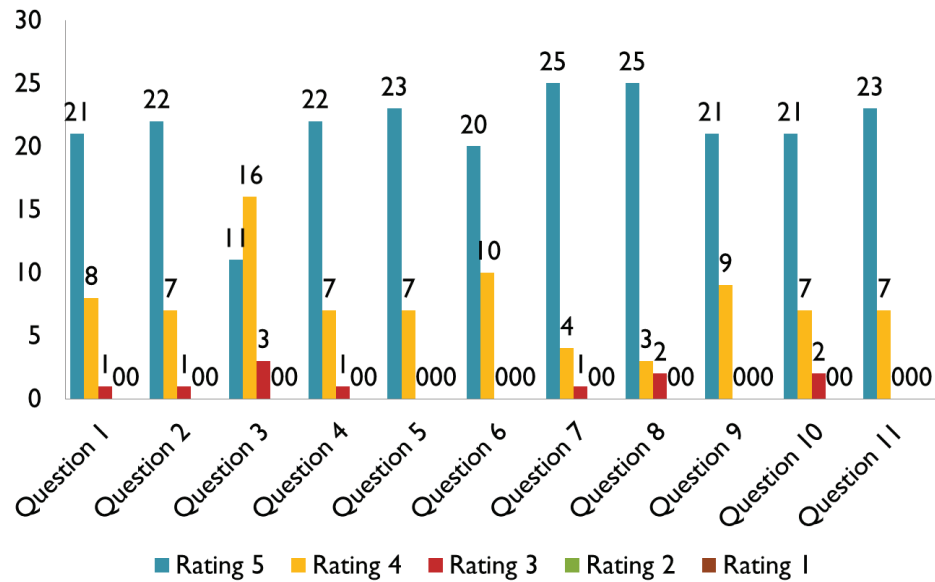
The system was built based on the architecture of Legal TrUTHS a similar system to IEfeL extracting information from legal Documents retrieved online. The architecture of Legal TrUTHS was based on Hobbs architecture of a generic Information Extraction system (Cheng et. Al., 2008). Legal TrUTHS and IEfeL are both domain dependent. The system does not need all the modules in Legal TrUTHS however it needed some additional modules and some modules were relocated and renamed and some functions of modules were modified a little.

IEfeL functions fully even if the filter was removed. Since the documents of the blue ribbon committee were very pattern based, it would just be a waste of computational expense if it would be invoked because the exact locations of the important keywords could be pinpointed without going through the dialogues in some of the documents which is what the filter has been designed to do. The system could function fully even if the Text Zoner was removed. The Text Zoner has also been removed because the system processes different types of documents whose headers and footers vary, the system already processes the documents fully so it would be irrelevant to place a Text Zoner, it would only add up to the computational complexity. The Semantic Tagger and Template Filler module of IEfeL acts basically the same way as the semantic interpretation and template generator module of Legal TrUTHS respectively does. A cross-referencing module has also been added to the system. Refer to 5.3.6 for the detailed description of the module. Basically, what it does is to identify which terms are synonymous to one another and register them to the database to be used by the front-end in searching. The CoReference module was retained because although it is rarely going to be used, sometimes pronouns occur in some entries to the template filler. Although these are not going to affect what would be entered to the field where the pronoun was found, the output would still appear in the tags so that the equivalent of the pronoun could still be associated to the document and thus used as a keyword for searching for that document. The CoReference module was improved further such that it could perform cataphora resolution on the document where cataphoric pronouns are evidently found.

6.2 Front-End Evaluation

Figure 6.1 shows the output of the survey that the research used to evaluate our system. The research asked 30 people to evaluate our front-end which is the website of the myDMS. The background of the people that evaluated our system are students, the research chose students because they might need it for school or maybe they wanted to know about the government right now. Another reason would be students are the future of our society, they will be the one that will be handling the society, so the research hopes that this website would be able to attract them to be more attentive with the government's issues. The process used in evaluating the website is by letting them see how the website works, how to use the system is also taught, after telling them, they have a choice to browse it by themselves or rate it with the use of first impression. The research then gives them evaluation forms which lets them rate on how the website is, in general, content and navigation wise.

Figure 6.1: Analysis of the Website



Questions used:

1. The website is easy to use
2. The design of the website
3. The website is interesting for normal users
4. The content is written clearly
5. The content is written in simple language
6. The content is easy to understand
7. The search results given by the website is sufficient
8. I found what I was looking for quickly
9. The website is easy to navigate around
10. The buttons explains clearly what will be next when clicked
11. The website responds quickly

In checking the total rating of the evaluation, many of our evaluators are satisfied with the Front-End module of the system. The rating is not lower than 3 which is neutral. The neutral rating only consists of a portion of the whole rating of the module. For some that was rated as neutral, one would be that the website is easy to use, one of the evaluators said that some of the buttons are not easy to understand and they are not sure when they should click a link

or a button. Another one would be the design of the website, an evaluator also said that the website is kind of cramped with all the links and buttons. Next one is, the website is interesting to normal users, it does not attract much of the normal users because most people right now are not quite familiar with our government. Next one would be the content is written clearly, the search results given by the website is sufficient, found what is being searched for easily and the buttons explains clearly what will be next when clicked. For the content of the result, few of the evaluators said that the output is somewhat cramped and the values are too many that it makes the window kind of dirty as for the navigation as mentioned earlier, the evaluators were not sure when to click a button or a link and there are too many buttons in the interface which makes it hard to choose what to do next. With the third question, "The website is interesting for normal users", the highest rating was only 4, the reason for this is the evaluators that the research asked were not really knowledgeable about the government and the website is too formal for them.

7 Conclusion and Recommendation

7.1 Conclusion

Overall, the system is meeting the standards requested by the Blue Ribbon Committee, with the help of Blue Ribbon Committee, the research was able to gather documents that would help in creating the system. The output of the system is already created and extracted based on the preference of the client.

The objectives that were achieved by the research are:

- Open and read Microsoft Word documents using Apache's POI hwpf library.
- Preprocess the document such as identify sentence boundaries and apply POS Tagging using LingPipe's API.
- Identify and normalize basic entities in order to comply with the designated standard format;
- Discard information that are most likely irrelevant;
- Use Anaphora Resolution to resolve issues in key entities using Ruslan Mitkov's algorithm;
- Construct a database consisting of tables that will hold specific filled-up templates which contains the information from the input document;
- Evaluate the performance of the Information Extraction system using Precision, Recall and F-measure;
- Detect synonyms of query and return probable related documents to the interest of the user;

With the help and constant communication with the Blue Ribbon Committee(BRC), the system has been built according to what is suited for them. Upon presenting the system to BRC and they remarked that what they had seen was beyond their expectations and they were very pleased about the output. New rules and heuristics were made by the research to build the Semantic Tagger, Cross Reference and Template Filler. The algorithm and rules used by the Template Filler and Semantic Tagger had been made by the research to fully support the documents of the Blue Ribbon Committee. There were also some additions to Ruslan Mitkov's rules so that the CoReference Resolution module would have an increase in accuracy despite its issues. The research built on top of the algorithm of Schwartz and Hearst for it to be used to build a table of synonymous names in the database which would be used by the Front-End module. Other than that, the research also implemented referencing in the Front-End module which would enable synonymous keywords to also be searched for in the database. The research also implemented a search ranking algorithm to be able to arrange the search output according to its relevance to the search key entered by the user. The algorithm is implemented by utilizing the IMAP engine of SQL which has fulltext matching that was used to score the search results, in addition, web crawler was also implemented to show relevant articles from "google news" and blogs or comments from "boardreader.com" which are related to the search keyword. The research envisions the use of innovation to enforce transparency

and could possibly lead to good governance by making documents readily available to the citizenry. In the same manner, the use of innovation by the citizens empowers them and would hopefully result to active participation and renewed trust to the government.

7.2 Recommendation

IEfeL was able to meet the objectives stated by the research. The research was able to identify issues which they recommend to be further worked on by researches aiming to build on top of IEfeL:

- Better Optical Character Recognition and a document reader that uses could be used so that the soft copy no longer needs to be uploaded and to reduce the steps in uploading the documents.
- The POS tagger of the system was trained using the brown corpus which LingPipe set as its default corpus. Due to this there were some incorrect tags that the POS Tagger module produced. In the future, blue ribbon committee documents could be used as its training documents to fit the domain better.
- The NER module is still dependent on the entities to be fed manually to the system. A recommendation on this module would be to use an NER algorithm which learns so that the manual input would no longer be used making the process shorter. The NER module of ANNIE is an alternative.
- The API used for reading the word documents, Apaches POI hwpf library is still underdeveloped. Since its closed source, updating this when the library can process the bullets in the document would pave way for the possible increase in the coreference module because this affects the paragraph number and sentence number affecting the gathering of candidates of the pronoun. It would also be a great help to the Cross Reference module as the retrieval of the persons department and position would be more accurate because the pattern is dependent on the bullet levels defined in the Agenda document.
- For the Semantic Tagger, a recommendation is to have the system to be able to process additional rules to be fed externally via a specification file. This is recommended so that it would be easier to add types of document that could be processed by the system.
- Transcript of Records could also be processed if algorithms and tools able to process sentences in a mixture of Filipino and English were used in the system.
- Additional templates could also be solicited so that more information relevant to the supposed users would be associated to the document.
- Image recognition algorithms could be applied to satisfy the request of the senate which is to be able to extract who among the senators signed the document.
- Update the web crawler to accommodate larger number of sources.
- Improve search ranking algorithm to provide more accurate scoring of results which will be based on the keyword searched by the previous users.

References

- Aarsvold, N. (1996). *Basic rules — demonstrative pronouns*. Available from <http://www.stolaf.edu/depts/norwegian/grammar/demonstratives.html>
- Alias-I. (2009). Lingpipe: Part-of-speech tutorial [Computer software manual]. Available from <http://alias-i.com/lingpipe/demos/tutorial/posTags/read-me.html>
- Appelt, D., Hobbs, J., Bear, J., Israel, D., Kamayama, M., & Tyson, M. (2004, February 24). The sri muc-5 jv-fastus information extraction system. Available from <http://www.isi.edu/~hobbs/muc5-fastus.pdf>
- Baldwin, B., & Carpenter, B. (2004). *Lingpipe*. Available from <http://www.alias-i.com>
- Boyle, R. (n.d.). *Hidden markov models*. Available from <http://www.comp.leeds.ac.uk/roger/HiddenMarkovModels/htmldev/main.html>
- Carpenter, B. (2006, July). Character language models for chinese word segmentation and named entity recognition. In *Proceedings of the fifth sishan workshop on chinese language processing* (pp. 169–172). Sydney, Australia: Association for Computational Linguistics. Available from <http://www.aclweb.org/anthology-new/W/W06/W06-0129.bib>
- Cheng, T. T., Cua, J. L., Tan, M. D., & Yao, K. G. (2008). *Legal truths (turning unstructured texts to helpful structures)*. Unpublished master's thesis, De La Salle University-Manila.
- Commission on information and communications technology - programs and projects*. (2009).
- Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V., Dimitrov, M., Dowman, M., et al. (2009, November). Developing language processing components with gate version 5 (a user guide) [Computer software manual].
- Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V., Ursu, C., Dimitrov, M., et al. (n.d.). Gate: A framework and graphical development environment for robust nlp tools and applications. In *In proceedings of the 40th anniversary meeting of the association for computational linguistics*.
- De Haan, P. (1989). *Postmodifying clauses in the english noun phrase: a corpus-based study*.
- Ekeklint, S. (2001). *Semantic tagging*. Available from <http://www.msi.vxu.se/~sek/articles/semtag.pdf>
- Elmi, M. A., & Evens, M. (1998). Spelling correction using context. In *Proceedings of the 17th international conference on computational linguistics* (pp. 360–364). Morristown, NJ, USA: Association for Computational Linguistics. Available from <http://dx.doi.org/10.3115/980845.980906>
- Emery, D. (1961). *Sentence analysis*. Holt, Rinehart and Winston Inc.
- Etzioni, O., Cafarella, M., Downey, D., Kok, S., Popescu, A., Shaked, T., et al. (2004). Web-scale information extraction in knowitall: (preliminary results). In *proceedings of the 13th international conference on world wide web*, 100–110. Available from <http://portal.acm.org/citation.cfm?coll=GUIDE38;dl=GUIDE38;id=988687>
- Giuliano, C., Lavelli, A., & Romano, L. (2006). Simple information extraction (sie): A portable and effective ie system. Available from <http://tcc.itc.it/people/lavelli/papers/giuliano-ATEM2006.pdf>
- Golding, A. R., & Schabes, Y. (1996). Combining trigram-based and feature-based methods for context-sensitive spelling correction. In *Proceedings of the 34th annual meeting on association for computational linguistics* (pp. 71–78). Morristown, NJ, USA: Association for Computational Linguistics. Available from <http://dx.doi.org/10.3115/981863.981873>
- Hobbs, J. R. (1993). The generic information extraction system. In *Muc5 '93: Proceedings of the 5th conference on message understanding* (pp. 87–91). Morristown, NJ, USA: Association for Computational Linguistics. Available from <http://dx.doi.org/10.3115/1072017.1072029>

- Jacobs, R. (1993). *English syntax a grammar for english language professionals* (R. Falk & A. Cooper, Eds.). Oxford University Press.
- Jayram, T. S., Krishnamurthy, R., Raghavan, S., Vaithyanathan, S., & Zhu, H. (2006). Avatar information extraction system. Available from <http://www.almaden.ibm.com/cs/projects/avatar/>
- Lappin, S., & Leass, H. J. (1994). An algorithm for pronominal anaphora resolution. *Comput. Linguist.*, 20(4), 535–561. Available from <http://portal.acm.org/citation.cfm?id=203987.203989>
- Lopresti, D. (2008). Optical character recognition errors and their effects on natural language processing. In *And '08: Proceedings of the second workshop on analytics for noisy unstructured text data* (pp. 9–16). New York, NY, USA: ACM. Available from <http://dx.doi.org/10.1145/1390749.1390753>
- Lyle, B. (2009). *Java api for wordnet searching(jaws)*.
- Makhoul, J. F. (1999, February). Performance measures for information extraction. in: *Proceedings of darpa broadcast news workshop*.
- Maynard, D., & Cunningham, H. (2003). Multilingual adaptations of annie, a reusable information extraction tool. In *Eacl '03: Proceedings of the tenth conference on european chapter of the association for computational linguistics* (pp. 219–222). Morristown, NJ, USA: Association for Computational Linguistics. Available from <http://dx.doi.org/10.3115/1067737.1067789>
- Mccallum, A. (2005, November). Information extraction: Distilling structured data from unstructured text. *Queue*, 3(9), 48–57. Available from <http://dx.doi.org/10.1145/1105664.1105679>
- Mitkov, R. (1998). Robust pronoun resolution with limited knowledge. In *Acl-36: Proceedings of the 36th annual meeting of the association for computational linguistics and 17th international conference on computational linguistics* (pp. 869–875). Morristown, NJ, USA: Association for Computational Linguistics. Available from <http://dx.doi.org/10.3115/980691.980712>
- Moens, M. F. (2006). *Information extraction: algorithms and prospects in a retrieval context*. Springer.
- Nadeau, D. (2005).
 balie - baseline information extraction
 . Available from <http://balie.sourceforge.net/dnadeau05balie.pdf>
- Nadeau, D. (2007). Semi-supervised named entity recognition: Learning to recognize 100 entity types with little supervision. Available from <http://cogprints.org/5859/1/Thesis-David-Nadeau.pdf>
- Nist/sematech e-handbook of statistical methods*. (n.d.).
- Ocr(technology). (2010). *Encyclopedia Britannica*. Available from <http://www.britannica.com/EBchecked/topic/430371/OCR>
- Park, Y., & Byrd, R. J. (2001). Hybrid textmining for finding abbreviations and their definitions. Available from <http://www.aclweb.org/anthology-new/E/E06/E06-2021.pdf>
- Radford, A. (2004). *Minimalist syntax* (S. R. Anderson et al., Eds.). Cambridge University Press.
- Ruch, P. (2002). Using contextual spelling correction to improve retrieval effectiveness in degraded text collections. In *Proceedings of the 19th international conference on computational linguistics* (pp. 1–7). Morristown, NJ, USA: Association for Computational Linguistics. Available from <http://dx.doi.org/10.3115/1072228.1072337>
- Sayed, I. Q. (2003). *Issues in anaphora resolution*. Available from http://nlp.stanford.edu/courses/cs224n/2003/fp/iqsayed/project_report.pdf
- Schwartz, A., & Hearst, M. (2003). A simple algorithm for identifying abbreviation def-

- initions in biomedical text. *Pacific Symposium on Biocomputing*. Available from <http://helix-web.stanford.edu/psb03/schwartz.pdf>
- Stevenson, M. (1987). *English syntax*. Little, Brown & Company(Canada) Limited.
- SunMicrosystems. (2008). *Mysql :: Why mysql*. Available from <http://dev.mysql.com/doc/refman/5.0/en/what-is-mysql.html>
- Toutanova, K., & Moore, R. (2002, July). Pronunciation modeling for improved spelling correction. In *Proceedings of the 40th annual meeting of the association for computational linguistics* (pp. 144–151). Available from <http://nlp.stanford.edu/kristina/papers/P02-1019.pdf>
- TrileX Labs. (2008). *Trilex labs - mydms*. Available from <http://www.trilexnet.com/>
- University, P. (2010). Wordnet, a lexical database for english.
- Warakagoda, N. (1998). *Definition of hidden markov models*. Available from <http://jedlik.phy.bme.hu/~gerjanos/HMM/node4.html>
- Yamada, T., Sakano, D., Yasumura, Y., & Uehara, K. (2006). Extraction of reliable reputation information using contributor's stance. *Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence*, 382–385. Available from Available online: <http://portal.acm.org/citation.cfm?id=124915438>
- Zhong, P., & Zhou, M. (2006). A generalized hidden markov model approach for web information extraction. In *proceedings of the 2006 IEEE/WIC/ACM international conference on web intelligence*, 709–718. Available from <http://portal.acm.org/citation.cfm?id=124906838>

A Part-of-Speech Tags used in the Hepple Tagger

CC - coordinating conjunction: "and", "but", "nor", "or", "yet", plus, minus, less, times (multiplication), over (division). Also "for" (because) and "so" (i.e., "so that").

CD - cardinal number

DT - determiner: Articles including "a", "an", "every", "no", "the", "another", "any", "some", "those".

EX - existential there: Unstressed "there" that triggers inversion of the inflected verb and the logical subject; "There was a party in progress".

FW - foreign word

IN - preposition or subordinating conjunction

JJ - adjective: Hyphenated compounds that are used as modifiers; happy-go-lucky.

JJR - adjective - comparative: Adjectives with the comparative ending "-er" and a comparative meaning. Sometimes "more" and "less".

JJS - adjective - superlative: Adjectives with the superlative ending "-est" (and "worst"). Sometimes "most" and "least".

JJSS - -unknown-, but probably a variant of JJS

-LRB- - -unknown-

LS - list item marker: Numbers and letters used as identifiers of items in a list.

MD - modal: All verbs that don't take an "-s" ending in the third person singular present: "can", "could", "dare", "may", "might", "must", "ought", "shall", "should", "will", "would".

NN - noun - singular or mass

NNP - proper noun - singular: All words in names usually are capitalized but titles might not be.

NNPS - proper noun - plural: All words in names usually are capitalized but titles might not be.

NNS - noun - plural

NP - proper noun - singular

NPS - proper noun - plural

PDT - predeterminer: Determinerlike elements preceding an article or possessive pronoun; "all/PDT his marbles", "quite/PDT a mess".

POS - possessive ending: Nouns ending in "'s" or ""'".

PP - personal pronoun

PRPR\$ - unknown-, but probably possessive pronoun

PRP - unknown-, but probably possessive pronoun

PRP\$ - unknown, but probably possessive pronoun, such as "my", "your", "his", "his", "its", "one's", "our", and "their".

RB - adverb: most words ending in "-ly". Also "quite", "too", "very", "enough", "indeed", "not", "-n't", and "never".

RBR - adverb - comparative: adverbs ending with "-er" with a comparative meaning.

RBS - adverb - superlative

RP - particle: Mostly monosyllabic words that also double as directional adverbs.

STAART - start state marker (used internally)

SYM - symbol: technical symbols or expressions that aren't English words.

TO - literal to

UH - interjection: Such as "my", "oh", "please", "uh", "well", "yes".

VBD - verb - past tense: includes conditional form of the verb "to be"; "If I were/VBD rich...".

VBG - verb - gerund or present participle

VBN - verb - past participle

VBP - verb - non-3rd person singular present

VB - verb - base form: subsumes imperatives, infinitives and subjunctives.

VBZ - verb - 3rd person singular present

WDT - wh-determiner

WP\$ - possessive wh-pronoun: includes "whose"

WP - wh-pronoun: includes "what", "who", and "whom".

WRB - wh-adverb: includes "how", "where", "why". Includes "when" when used in a temporal sense.

:: - literal colon

, - literal comma

\$ - literal dollar sign

- - literal double-dash

- literal double quotes

- literal grave

(- literal left parenthesis

. - literal period

- literal pound sign

) - literal right parenthesis

- literal single quote or apostrophe

B Sample Documents from the Blue Ribbon Committee

The following are sample documents from the Blue Ribbon Committee.



REPUBLIC OF THE PHILIPPINES
CONGRESS OF THE PHILIPPINES
SENATE
Pasay City

MEMORANDUM:

Date : October 11, 2007
For : SENATE PRESIDENT MANNY B. VILLAR
From : JOSEPHINE G. HERRERA
Subject : SPOT REPORT ON THE INITIAL PUBLIC HEARING OF THE COMMITTEE ON ACCOUNTABILITY OF PUBLIC OFFICERS AND INVESTIGATION (BLUE RIBBON) JOINT WITH COMMITTEES ON TRADE AND COMMERCE, AND NATIONAL DEFENSE AND SECURITY RE:

P.S. Res. No. 127 - "RESOLUTION DIRECTING THE BLUE RIBBON COMMITTEE AND THE COMMITTEE ON TRADE AND INDUSTRY TO INVESTIGATE, IN AID OF LEGISLATION, THE CIRCUMSTANCES LEADING TO APPROVAL OF THE BROADBAND C WITH THE ZTE AND THE ROLE OF THE OFFICIALS CONCERNED IN GETTING IT CONSUMMATED, AND TO MAKE RECOMMENDATIONS TO HALE TO THE COURTS OF LAW, THE PERSONS RESPONSIBLE FOR ANY ANOMALY IN CONNECTION THEREWITH AND TO PLUG LOOPHOLES, IF ANY, IN THE BOT LAW AND OTHER PERTINENT LEGISLATION (Introduced by Senator Aquilino Q. Pimentel, Jr.)" *Hearing Highlights / Spot*

P.S. Res. No 129 - "RESOLUTION DIRECTING THE COMMITTEE ON NATIONAL DEFENSE AND SECURITY TO CONDUCT AN INQUIRY IN AID OF LEGISLATION INTO THE NATIONAL SECURITY IMPLICATIONS OF AWARDED THE NATIONAL BROADBAND NETWORK CONTRACT TO THE CHINESE FIRM ZHONG XING TELECOMMUNICATIONS EQUIPMENT

COMPANY LIMITED (ZTE CORPORATION),
WITH THE END IN VIEW OF NATIONAL
SOVEREIGNTY AND TERRITORIAL
INTEGRITY" (Introduced by Senator Panfilo M.
Lacson).

Privilege Speech of Senator Panfilo M. Lacson
entitled "Legacy of Corruption" delivered on 11
September 2007.

Call to order: 10:53 a.m. Suspension: 2:30 p.m.

MEMBERS/SENATORS PRESENT: Senators Alan Peter S. Cayetano
(Chair, Blue Ribbon), Manuel A. Roxas (Chair, Committee on
Commerce and Trade), Rodolfo G. Biazon (Chair, Committee on
National Defense and Security), Jinggoy Ejercito-Estrada (President
Pro-Tempore and Ex-Officio Member), Panfilo M. Lacson, Juan Ponce
-Enrile, Francis G. Escudero, Aquilino Q. Pimentel, Jr. (Minority
Leader and Ex-Officio Member), Joker P. Arroyo, Francis N.
Pangilinan (Majority Leader and Ex-Officio Member), Benigno Simeon
C. Aquino III, Miguel F. Zubiri, Loren B. Legarda, Jamby A.S.
Madrigal, Pia S. Cayetano.

GUESTS/RESOURCE PERSONS: Mr. Jose de Vencia III (Amsterdam
Holdings, Inc), Mr. Ramon P. Sales (Former Chair, Commission on
Information and Communication Technology), Vice Go. Rolex T.
Suplico (Iloilo), and Mr. Jarius Bondoc (Columnist, The Philippine
Star).

DISCUSSIONS

A joint public hearing was held by the Committees on Accountability
of Public Officers and Investigations, Trade and Commerce, and National
Defense and Security to consider the following: 1) a resolution which seeks
to investigate the approval of the broadband contract with ZTE; 2) a
resolution directing the Committee on National Defense and Security to
conduct an inquiry into the national security implication of awarding the
National Broadband Network contract to ZTE; and 3) privilege speech of
Senator Lacson entitled; "Legacy of Corruption".

Printed and released by Blue Ribbon Committee on Transparency & Accountability

Figure B.2: Hearing Highlights Document Page 2

Senator Enrile inquired into the enforceability of the National Broadband Network (NBN) Contract. He opined that the document presented is not yet a contract. He believes that such is not yet an enforceable contract and therefore the government is not bound since the terms and conditions have not been met yet. Senator Alan Cayetano explained that the Committee would like to look into the process and circumstances leading to the contract.

Senator Escudero requested the Committee to issue *subpoena duces tecum* with respect to the documents mentioned in the contract submitted by the DOTC, namely: attachments of paragraph No. 41.6 contained in pages 34 and 35; obligational authority issued by DBM, loan agreement between X bank and DOP; legal opinion of DOJ; executive agreement between the Republic of the Philippines and the People's Republic of China; letter of Chinese Ambassador Li Jin Jun; letter of NEIDA Secretary to Minister Bo Xilai; and DOTC's evidence of authority to represent the Philippine government.

Senator Alan Cayetano stated that the inability of Secretary Neri to attend the hearing was due to a prior invitation of the Committee on Finance for the budget hearing and in compliance with EO 464. Senator Pimentel suggested to post OSAA guards at the doors of the committee room and to bring Mr. Neri to the committee hearing after the CHED budget hearing. Senator Enrile objected to the suggestion of Senator Pimentel since there had as yet been no compulsory process served on Chairman Neri. Senator Pimentel then suggested to the Committee to issue a subpoena immediately, to prevent Chairman Neri from leaving the Senate premises. However, the Chair informed the Committee that CHED Chairman had left the building. Senator Pangilinan then recommended to proceed with the issuance of subpoena to all resource persons who failed to appear in the hearing.

The highlights of Jose De Venecia III's affidavit are as follows: In December 2006, Amsterdam Holding Inc. (AHI) filed its unsolicited proposal for NBN Project in accordance with the BOT Law; Chairman Abalos approached and offered him a technical partnership with ZTE in exchange for ten million dollars; He found out that the ZTE's proposal was overpriced by 100% or 130 million dollars to accommodate advances and kickbacks; When he joined Chairman Abalos in Shenzhen China to meet with ZTE officials the Chairman demanded from ZTE the balance of his commission for such are due to the President and the Speaker; Upon discovery of the bribe, he had an overwhelming reservations against ZTE's proposal because it will incur more debt for the country; His company filed a petition before the Supreme Court to compel the DOTC to reveal the terms of ZTE contract; Chairman Abalos threatened him and Mr. Bondoc's life; In a reconciliatory meeting, PG Mike Arroyo pointed a finger at him and yelled "back-off", which he took to mean as having to withdraw from the NBN project.

Figure B.3: Hearing Highlights Document Page 3

Senator Estrada inquired whether the contract requires sovereign guarantee, De Venecia replied in the affirmative. He also asked who were the other personalities that were present in the reconciliatory meeting. De Venecia replied that Secretary Mendoza, Chairman Abalos, Jimmy Paz, Quirino Dula Torre, Police General Ruben Reyes, and Leo San Miguel were present. De Venecia further said that he believes that Chairman Abalos requested the First Gentleman to ask him to withdraw from the project.

Senator Laeson asked whether the First Gentleman was present in any of the meetings with ZTE officials, De Venecia replied that he does not know because he was never present in any of the meetings with ZTE officials. Senator Laeson presented two pictures to De Venecia and asked to identify Edgar Dela Torre. De Venecia identified the picture of Quirino Dela Torre and not Edgar Dula Torres as being present.

Senator Laeson noted that ZTE's draft proposal amounted to \$262 million and which would only cover 30% of the entire country compared to the final Supply Contract that was signed by Secretary Mendoza which amounted to \$329,481,290.

Senator Laeson further stated that on August 25, 2007, there was a loan agreement signed by Secretaries Teves, Pavila and Mendoza, for the Philippines and Commerce Minister Bo Xilai, for China. He requested the Committee to issue a *subpoena duces tecum* for the production of the loan agreement.

As an administrative matter, Senator Escudero requested that corresponding invitations be issued to Mr. Ruben Reyes, Mr. Jimmy Paz, Gen. Quirino Dula Torre, and Leo San Miguel. Senator Pimentel suggested that a subpoena be issued instead of an invitation. On the other hand, Senator Arroyo objected to the suggestion of issuing a subpoena because of the need to be very careful about the rights of witnesses in any proceedings. He further stated that the Senate must follow its own rules. However, the Chair disagreed with the position of Senator Arroyo that an invitation must first be issued before a subpoena because it is not within the rules of the Senate. Senator Arroyo then expressed his views on the rules that an invitation must first be issued, followed by subpoena. If despite the invitations and subpoenas, the witnesses fail to appear, the Senate can issue a contempt citation.

Senator Escudero asked De Venecia to submit to the Committee the unsolicited proposal of AHL. He also asked if the President is involved in the said contract and if it is within the power of the President to rescind the contract. De Venecia said that the President wants the contract to be reviewed and evaluated. Senator Cayetano also broached Sec. 5 of RA 3019 that prohibits the spouse or relative by consanguinity or affinity of the President, Vice-President, Senate President or the Speaker of the House from intervening in any government transaction.

Report filed pursuant to the Freedom of Information Act (RA 3019) - Transparency Section

Figure B.4: Hearing Highlights Document Page 4

On the query of Senator Pangilinan as to the possible involvement of the President, De Venecia replied that based on his personal knowledge the President is not involved in the controversy. Senator Pangilinan further raised that \$10 million is a huge amount to be refused. Mr. De Venecia emphasized that he does not want to be associated with any bribery or acceptance of any amount from any public officials.

Mr. De Venecia again recalled that during their meeting with ZTE executive in Shenzhen China, Chairman Abalos asked for the balance of his (Abalos) commission because he would be busy in the upcoming election. Vice-President Yu Yong of ZTE refused to give more money until the loan documents are signed.

The Chair inquired into the status of the "Telepono sa Barangay Project" where the government spent P10 to P20 billion. Mr. Sales replied that the project was overtaken by new technologies. In addition, Senator Biazon stressed the possibility that before the government can pay the 20-year loan agreement the system may already have become obsolete.

Senator Biazon requested that the annexes in the affidavit of Mr. Jose De Venecia be submitted to the Committee.

Senator Escudero pointed out that there was a study made by Professors Fabella and De Dios of University of the Philippines. He then requested that the two professors be invited to the hearing.

Senator Madrigal, for her part, stressed the following as conditions to justify a government broadband network: it must be undertaken by a build-operate-and transfer scheme; it must be paid for by private funding and not through government funds; there should be no government subsidy or outlay for the project; pay as use instead of "take or pay" scheme; and, the results of the undertaking is a reduction in government telecommunications expenses.

Mr. Sales claimed that NBN will be good for the country if the private sectors are the ones to undertake it. However, the private sectors seemed to be unsure in providing broadband network for the country.

Vice-Governor Suplico asserted that the agreement that was signed in China last April 21 is enforceable since all the requisites of a contract are present.

As to the proponent of the NBN project, Mr. Sales affirmed that it was conceived to be an integrated communication network for the government. The President mandated him to convert the government communications using VOIP. He added further that there was EO 603 that was issued to move Telof to DICT.


Prepared and submitted by: Blue Ribbon Oversight and Ethics Management - Investigations Service

Figure B.5: Hearing Highlights Document Page 5

Mr. Jarius Bondoc said that he attended a forum in Ateneo Professional School where Assistant Secretary Formoso presented an audio-visual presentation. Asec. Formoso admitted during the presentation that the government wants to save P4 billion pesos in annual telecom expenditure. He also admitted that the NBN project will not include cellular phones, cellular calls, services and internet services will have to come from the private sector. In such case, the government will have to appropriate annually funds for the operation of NBN.

The Chair then announced that there will be a continuation of the hearing on September 20, Thursday at 2:30 p.m. Other guests who failed to attend the hearing do not need an invitation, although the Secretariat was directed to issue subpoenas or invitations.

Respectfully Submitted:


JOSEPHINE G. HERRERA

Noted by:



ATTY. RODOLFO NOEL S. QUIMBO
Blue Ribbon Oversight Office Management
cc: All Senators

Figure B.6: Hearing Highlights Document Page 6

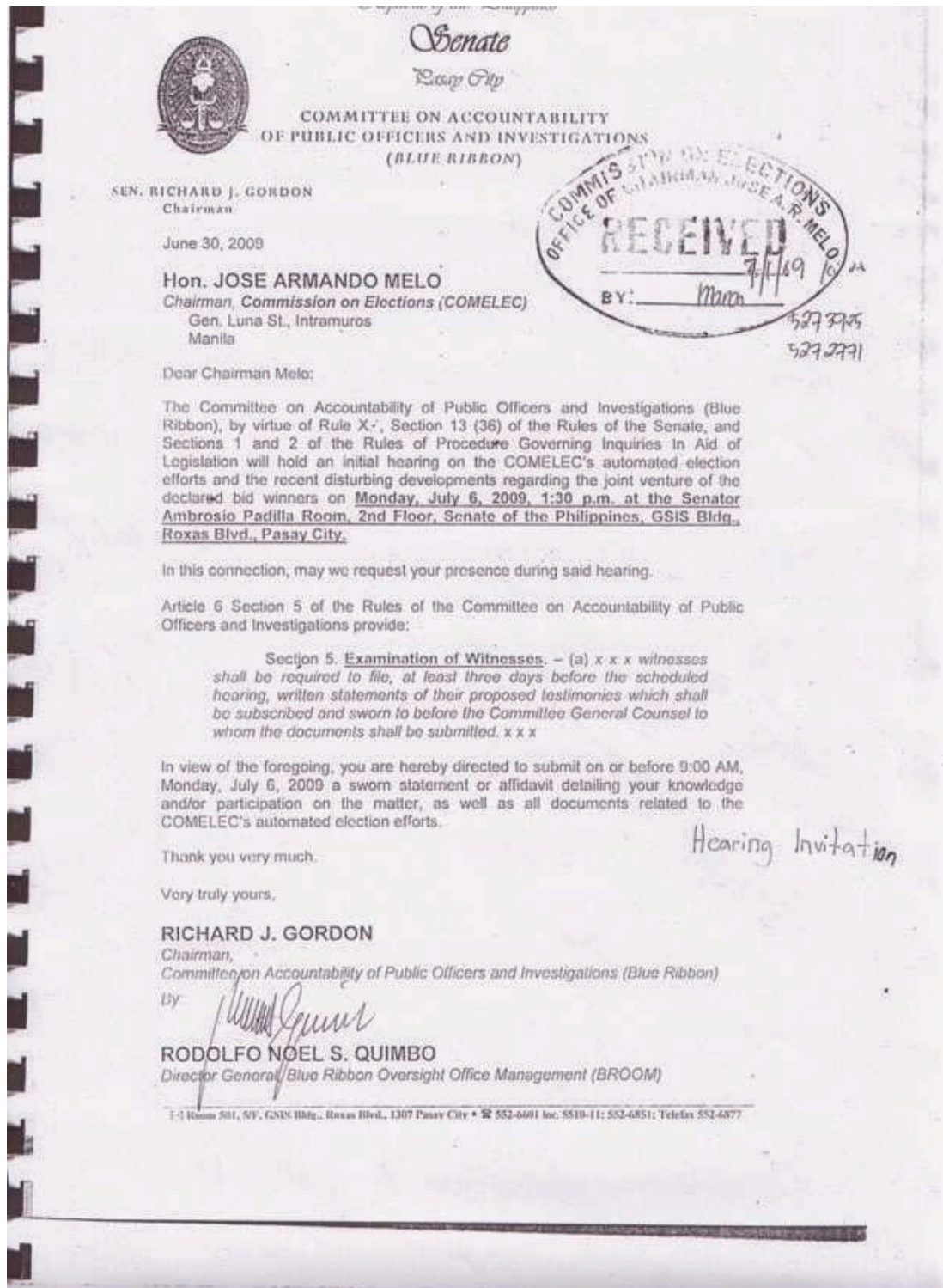


Figure B.7: Hearing Invitation Document

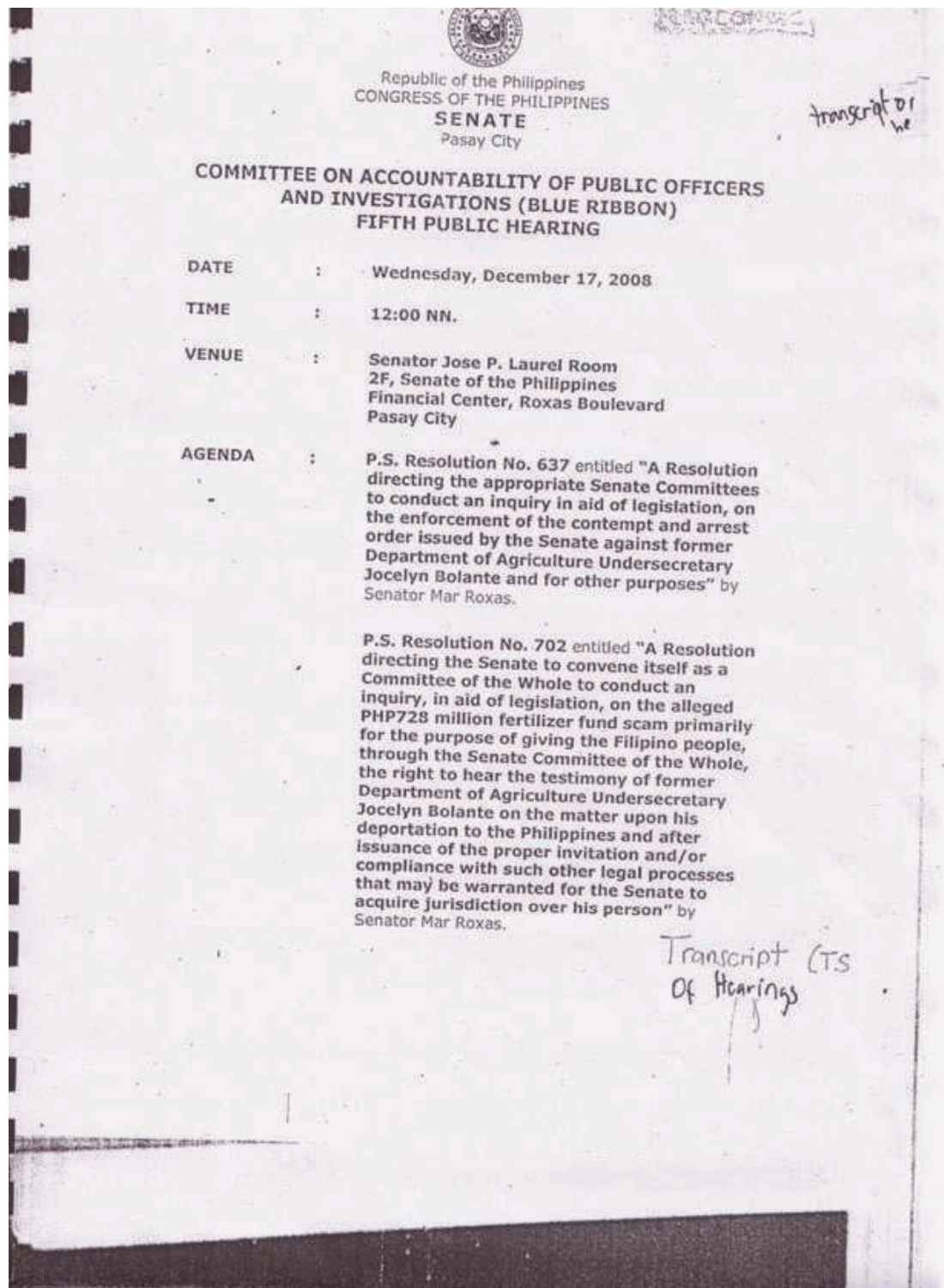



Figure B.8: Transcript of Hearings Document


Senate of the Philippines
Manila

Senate
Room

SENATE OF THE PHILIPPINES
RECORDS MANAGEMENT & MAILING SERVICE
RECEIVED
BY: Jeg M
DATE: 7/12/09 TIME: 8:20 AM

MEMORANDUM

To : Head of Office/Chief of Staff
Service Units/Office of Senators

From : Atty. Ma. Cynthia E. Bajamunde
Officer in Charge, Property and Procurement Service

Date : July 7, 2009


Subject : Inventory of All Office Supplies and other Consumable items

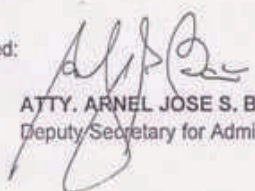
Please be advised that the "Supplies and Materials Unit of Property and Procurement Service" will conduct its semi-annual inventory of all office supplies and other consumable items as per *GAAM Section 490*, for the period of July 13, 2009 to July 21, 2009.

There will be no issuance of office supplies for the said period.

Issuance of supplies will resume on July 22, 2009.

For your information and guidance.



Noted: 

ATTY. ARNEL JOSE S. BAÑAS
Deputy Secretary for Administration & Finance

SENATE OF THE PHILIPPINES
RECORDS MANAGEMENT & MAILING SERVICE
SENATE Memorandum
Jm
7/12/09
DATE TIME

Figure B.9: Senate Memorandum Document

Republic of the Philippines)
Quezon City)S.S.

BLUE RIBBON COMMITTEE

RECEIVED BY, Amr 2/11/08
1.05 PM

AFFIDAVIT

I, **RODOLFO I. LOZADA Jr.**, of legal age, Filipino,
holding office at the Philippine Forest Corporation, Old
NAMRIA Building, Lawton Avenue, Fort Bonifacio, Taguig City,
being first duly sworn, hereby depose and state:

1.01 I am the Charter President (Founder) of the
Philippine Forest Corporation, a subsidiary of the Department
of Environment and Natural Resources ("DENR"). I have been
such Charter President since its inception on January 2005
and continuously up to the present.

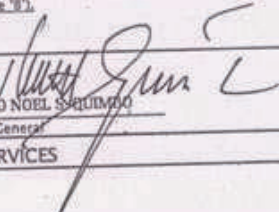
docu. evid

1.02 I am a professional ECE 2235 engineer, having
graduated with a degree in electronics and communications
engineering, which I obtained from the University of Santo
Tomas in 1984. I passed the government board examinations
for electrical engineers in 1985.

ELECTRONICS & COMMUNICATIONS.

Sample of Document
Evidence

Figure B.10: Document Evidence

JOB/SERVICE REQUEST FORM			
A. Requesting Unit: <u>BLUE RIBBON OVERSIGHT OFFICE</u> <u>MANAGEMENT (BROOM)</u>		Date: <u>26 AUGUST 2008</u>	
SERVICE REQUESTED <div style="display: flex; justify-content: space-between;"> <div style="width: 45%;"> <input type="checkbox"/> Carpentry <input type="checkbox"/> Electrical <input type="checkbox"/> Aircon <input checked="" type="checkbox"/> Telephone/Fax/Intercom <input type="checkbox"/> Others (Please specify) </div> <div style="width: 45%;"> <input type="checkbox"/> Sound System <input type="checkbox"/> Plumbing <input type="checkbox"/> Janitorial Services </div> </div>		DESCRIPTION/SPECIFICATION OF JOB/SERVICE REQUESTED: <u>Installation of telephone and telephone lines at the BROOM extension office (3rd Floor, Room 304, beside Committee 'B').</u>	
Requested By: <u>MELANIE U. DOMANGUAL</u> <u>BROOM Staff</u>		Noted By:  <u>RODOLFO NOEL S. BUMBAO</u> <u>Director General</u>	
B. To be filled-up by section under MPFS or GENERAL SERVICES			
Technical Inspection/Actual Job/Service Needed: 			
Specification of materials needed for the job, if any			
QTY.	UNIT	DESCRIPTION	ESTIMATED COST
TOTAL ESTIMATED COST			
RECOMMENDED ACTION Job Service Undertaken By: <div style="display: flex; justify-content: space-between;"> <div style="width: 45%;"> <input type="checkbox"/> MGSB Personnel (In-House) <input type="checkbox"/> Hired Labor <input type="checkbox"/> Outside Contractor </div> <div style="width: 45%;"> Remarks </div> </div>			
Inspected By: <u>Technician</u>		Certified By: <u>Division Chief</u>	
Approved By: <u>Director, MGSB</u>		 	
C. For Jobs/Services which require cost:			
Recommending Approval/Disapproval: <u>Dep. Sec Admin & Fin. SVS/Secretary</u>		Approved/Disapproved: <u>Secretary/Senate President</u>	
 		Date 	
D. Acknowledgement			
This is to acknowledge completion of the Job/Service requested by the _____ as of this date _____			
_____ CHIEF, REQUESTING UNIT			

SENATE OF THE PHILIPPINES
 DATE: 8/24/08
 RECEIVED
 BY: [Signature]
 MAINTENANCE GEN. SERVICES BUREAU

Other Administrative Documents (Request)

Figure B.11: Other Administrative Documents

C Sample Templates

The following are sample templates of the different documents that will be processed by the system.

Table C.1: Hearing Highlight sample template fields

Hearing Highlight	
Fields	Data
Date	
Receiver	
Sender	
Subject	
People involved / present	
Organizations involved	
Laws incorporated	
Important dates	

Table C.2: Hearing Invitation sample template fields

Hearing Invitation	
Fields	Data
Date	
Receiver	
Sender	
Organizations involved	
Date and time of the hearing	
Address of the hearing	

Table C.3: Senate Memorandum sample template fields

Senate Memorandum	
Fields	Data
Date	
Receiver	
Sender	
Subject	
Organizations involved	
Date and time of the hearing	

Table C.4: Document Evidence sample template fields

Document Evidence	
Fields	Data
Type of document	
Persons involved	
Details of the person	
Dates involved	

Table C.5: Referral sample template fields

Referrals	
Fields	Data
Sender	
Position of the Sender	
Receiver	
Position of the Receiver	
Subject	
Date sent	
Dates Involved	

Table C.6: Notice of Hearing sample template fields

Notice of Hearing	
Fields	Data
Sender	
Date	
Organizations Involved	
Members of each organization	
Guests	
Date and time of the hearing	
Address of the hearing	

Table C.7: Committee Report sample template fields

Committee Report	
Fields	Data
Committee Report No.	
Short Title	
Filed Date	
Committees which reported	
Bills or Resolutions	
Date, Time and Venue of Meetings	
Organizations involved	
Dates involved	
Committee reports involved	

D Resource Persons

Mr. Allan Borra

Thesis Adviser, Faculty Member
Software Technology Department, College of Computer Studies
De La Salle University-Manila
borgz.borra@delasalle.ph

Mr. Sherwin Ona

Faculty Member
Information Technology Department, College of Computer Studies
De La Salle University-Manila

Attorney Rodolfo Quimbo

Blue Ribbon Committee
Senate of the Philippines
rudy.quimbo@senate.gov.ph

Ms. Fritz Ambay

Blue Ribbon Committee
Senate of the Philippines
fritz_ambay@yahoo.com

Ms. Amor Abagon

Blue Ribbon Committee
Senate of the Philippines
amorabagon@yahoo.com

Sir Jayson Hecita

La Salle Institute of Governance
De La Salle University-Manila

Dr. Eric Vincent C. Batalla

Chair ,Psychology Department
De La Salle University-Manila

Mr. Rico Locsin

Political Student Leader
De La Salle University-Manila

E Personal Vitae

Mr. Brian Kent T. Lim

87 Don Ramon Street, St. Francis Village, Dolores, San Fernando, Pampanga, Philippines
(0926)6865951
brian_lim_14@yahoo.com

Mr. Angelo Crisanto B. Miranda

397-A Hipodromo Street, Sta. Mesa, Manila, Philippines
(0915)8239729
acme_miranda@yahoo.com

Ms. Janine D. Trogo

U1719 EGI Taft Tower 2339 Taft Avenue, Malate, Manila, Philippines
(0916)2507793
kaxelian@yahoo.com

Ms. Fe Eleanor L. Yap

Block 14 Lot 6 Willow Street, Victoneta North, Potrero, Malabon, Philippines
(0917)8500617
feyap1990@yahoo.com