References

[1] Adar E., Karger D., Stein L., (1999), *Haystack: Per User Information Environments*, In Proceedings of the 1999 Conference on Information and Knowledge Management, CIKM 1999

[2] Alani H., Kim S., (2005), *Automatic Ontology-based Knowledge Extraction and Tailored Biography Generation from the Web*, Intelligence, Agents, Multimedia Group University of Southampton

[3] Bordes J., Maitre H., (2008), *Automatic Semantic Network Construction for Multi-Layer Annotation of Satellite Images*, hal-00315531

[4] Carlson A., Schafer C., (2008), *Bootstrapping Information Extraction from Semi-structured Web Pages*, Machine Learning Department, Carnegie Mellon University, Pittsburgh

[5] Chia-Hui C., Chun-Nan H., Shao-Chu L., (2002), *Automatic Information Extraction from Semi-Structured Web Pages By Patern Discovery*, National Central University, Taiwan

[6] Davis R., Shrobe H.E., Szolovits P., (1993), *What is Knowledge Representation*, AI Magazine 14(1), PP 17-33

[7] Geng J., Yang J., (2004), *AUTOBIB: Automatic Extraction of Bibliographic Information on the Web*, Department of Computer Science, Duke University, Durham, NC 27708, USA

[8] Guo J., Chaisatien P., (2010), *Partial Information Extraction Approach to Lightweight Integration on the Web*, Department of Computer Science, Tokyo Institute of Technology, Japan

[9] Hannon, (2011), *Tips to Effective Internet Searching*, Available: http://hanlib.sou.edu/searchtools/searchtips.html

[10] Hedley Y., Younas M., James A., Sanderson M., (2004), In*formation Extraction from Template Generated Hidden Web Documents*, School of Mathematical and Information Sciences, Coventry University, UK

[11] Jain M., Mitra P., (2008), *A Generalized Knowledge Representation Architecture for Intelligent Tutoring System*, Indian Institute of Technology Kharagpur

[12] Krsul I., (2009), *Co-Word Analysis Tool,* Available: http://krsul.org

[13] Lam M.I., Gong Z., Muyeba M., (2008), *A method for Web Information Extraction,* Faculty of Science and Technology, University of Macau , Macao

[14] Merialdo V.C.G.M.P., (2001), *RoadRunner: Towards Automatic Data Extraction from Large Web Sites,* Proceedings of the 27th VLDB Conference, Roma, Italy

[15] Nikolas K.P., (2010), *STAVIES: A System for Information Extraction from unknown Web Data Sources through Automatic Web Wrapper Generation Using Clustering Techniques,* IEEE Transactions on Knowledge and Data Engineering

[16] Ozyer A.F., Alhajj T.R., (2008), *Employing Clustering Techniques for Automatic Information Extraction From HTML Documents*, Dept. of Computer Science, Calgary University, Calgary, AB

[17] Peters S. and Shrobe H.E., (2002), *Using Semantic Networks for Knowledge Representation in an Intelligent Environment,* MIT Artificial Intelligence Laboratory, Cambridge, MA, 02139, USA

[18] Qin H., (1999), *Knowledge Discovery Through Co-Word Analysis,* LIBRARY TRENDS, Vol. 48, No. 1, Summer 1999, PP 133-159

[19] ReadWriteWeb, (2011), *Semantic Web Difficulties With Classic Approach,* Available: http://www.readwriteweb.com/archives/ semantic_web_difficulties_with_classic_approach.php

[20] Richard V., Contreras J., (2010), *Six Challenges for the Semantic Web,* Intelligent Software Components, S.A

[21] Sánchez D., Moreno A., (2005), *Automatic information extraction from the Web,* Department of Computer Science and Mathematics ,Universitat Rovira i Virgili

[22] Thamvijit D., Chanlekha H., (2005), *Person Information Extraction from the Web* , Department of Computer Engineering, Faculty of Engineering, Kasetsart University , Bangkok, 10900, Thailand

[23] Tran-Le M.S., Vo-Dang T.T., Ho-Van Q., Dang T.K., (2008), *Automatic Information Extraction from the Web: An HMM-Based Approach,* Faculty of CSE, University of Technology, Ho Chi Minh City, Vietnam

[24] W3C, (2011), *Semantic Web,* Available: http://www.w3.org/standards/semanticweb

[25] Wang C.B., Yang H., (2006), *An Intelligent Web Knowledge Extraction Framework to Support E-Learning*, Kainan University, Taiwan

[26] Zhuge H., (2006), *Autonomous semantic link networking model for the Knowledge Grid,* Concurrency Computat.: Pract. Exper. 2007; 19, PP 1065–1085

Appendix A

Knowledge Extractor Software Design Details

Software design of knowledge extractor have been done using best practices in the software engineering discipline. We first sub divide the whole system into for sub systems namely Knowledge Extractor, Knowledge Base, Searcher and user interface. Here the Searcher stands for the inference engine. Each of the sub system consists of few classes depending on the responsibilities allocated to each sub system. Below we show the class diagram for the system in Figure A.1.

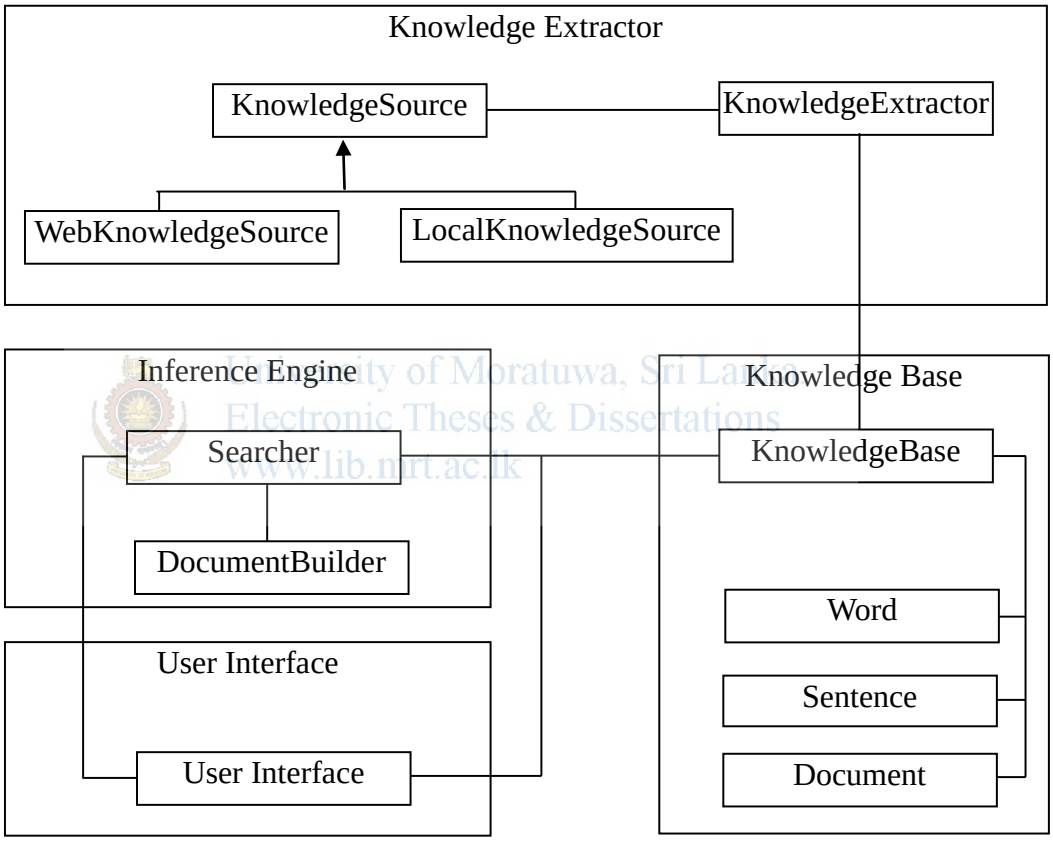

Figure A.1: Knowledge Extractor Class Diagram

In this design we have enabled support for multiple knowledge sources through the deriving each specific knowledge source class from the base KnowledgeSource class. The KnowledgeSource class provides the interface,

```
virtual void find(std::string) = 0;
```

so that other modules can request for more information. After a request received the knowledge source should fetch necessary information and remove all unwanted meta

data from the source document before handing over to other system.

All the gathered information in knowledge source is forwarded to the KnowledgeExtractor through a callback function,

```
virtual void onData(std::string ref, std::ifstream& in);
```

After this point the knowledge extractor performs all the statistical processing on the received data file and update the knowledge base.

Knowledge base has been implemented as a wrapper over a conventional server less database management system and provides intuitive knowledge engineering interface to other modules. The knowledge base interface is as follows.

```
void    init();
void    reset();

std::pair<int, bool>    addReference( std::string ref );

int     addSentence( std::string sentence, int ref );
int     addWord( std::string word, int sentence );
void    addCoWords( int, int );

int                     getWordId( std::string word );
std::string             getWord( int wordId );
int                     getWordAppearances( int wordId );
std::set<int>           getCoWords( int wordId );
int                     getCoWordAppearances( int wordId1, int wordId2 );
std::set<int>           getContainingSentences( int wordId );
Sentence                getSentence( int sentenceId );
std::set<int>           getContainingReferences( int sentenceId );
std::string             getReference( int referenceId );
```


Searcher sub system acts as the inference engine of the system. It accepts user queries through the following interface and returns a document as the output of the query.

```
std::string search(std::string sentence);
```

The returned string contains the name of the output document in the file system. User interface uses this file name to display the output. Searcher delegates document building part to the DocumentBuilder instance. It formats its output as a HTML document so that the user interface can use WebView control to easily display the content.