Locating Services in Legacy Software:Information Retrieval Techniques, Ontologyand FCA Based Approach

MOSTEFAI ABDELKADERMALKI MIMOUNBOUDCHIHA DJELOUL
Department of Mathematics and Computer Science,
D[r]TaharMoulay, Saida University
D[r]TaharMoulay University, City Ennasr, Saida20000, Algeria
Tel/Fax. : +213 (0) 48.47 42 62
Emails :mostefaia_aek@yahoo.fr, Malki_m@yahoo.fr, Boudj@yahoo.fr

Abstract: -The localisation and identification of services in legacy software is the most challenging task in the process of Migrating (i.e. reengineering) legacy software towards service oriented architectures (i.e. SOA) and web services technologies. This paper propose an approach to locate services in legacy software by means of information retrieval techniques, WORDNET ontology, FCA (i.e.Formal Concepts Analysis) and the analysis of the legacy interfaces of the software.In this approach interfaces are analysed to generate queries for each service to be located , the WORDNET ontology is used to expand the queries terms to make best coverage of the Modules (e.g. the part of source code : procedures /functions)implied in the computation of the service,IR (i.e. Information Retrieval) techniques (e.g. vector space model,Latent semantic analysis) are used to map queries to the relevant modules of the legacy software,presented as ranked list (i.e. search space), this list represent the parts of source code that participate in the computation of the service.This process is repeated for each service and the results are exploited by the FCA techniques to reduce the search space time spent by developer when examining the result to decide in real parts that contribute to the computation of each service.

Key-Words:Migrating, Legacy software,FCA,Ontology,Information Retrieval Techniques, Interfaces,Web Services.

1 Introduction

SOA and web service technologies are new approaches to engineer interoperate software’s so that companies internal andexternal applications (i.e.Clients and vendors applications) can be integrated,reports publish that 50 % of the new engineered application are engineered according to the SOA paradigm [1]. .

Most companies rely on legacy softwarefunctionalities toaccomplish their daily business andreengineering these functionalities towards SOA approach is the objective of the enterprise. Reengineering the legacy software towards SOA architectures make it interoperable and add business value to it,so external (and internal) application can consume the provided service, reengineering is less costly then the redevelopment from scratch,reengineering provides a mean to companies to capitalize their anterior investments,in Chatarji [2] provide a set of advantages to migrate towards SOA architecture .

The first step needed to reengineer the legacy software is the identification (i.e.. location) in the source code of the functionalities to be exported (i.e. reused) as web services in the SOA paradigm [3,4,5,6,7]. Program understanding is the key of the success of this step.Locating services in source code can be taken as concept (i.e. feature) location problem with some assumption , concept location is the process of finding part of source code that participate in the computation of some functionalities[3]and is prerequisites of any program understanding task. Several approaches had been proposed to locate concept in source code ranging from static analysis of the source code [8] and the dynamic approaches[20],mixed between static and dynamic [10] and the approaches that make use of IR techniques [11].IR approaches [29] are good tools to find concept (i.e. service) but the main problematic (general problematic not only in the context of service location) is when writing query (i.e. choosing set of descriptive terms) to find a concept (e.g. service),the terms that compose the query can be part also of other services terms (i.e. the functionalities share terms) so the ranked result list of the mapping between query and source code parts (modules) in term of an IR technique is so large and mixed between set of part really participating in the computation of the concepts and part that do not, we name this problematic as overlapping queries terms effects.

As solution to this problem we propose an approach to locate services in legacy source code starting from the extraction of a specification (i.e. term inputs,terms outputs and name of the service) of each service to be located based on the interface of each legacy functionality, this specification is expanded using the WORDNET ontology, then each specification is mapped to service query. Each query is mapped to modules of source code and the result is ranked list of candidate relevant modules that probably contribute to the computation of the service.The set (queries, list of relevant modules) is used to build context table (objects are service to be located and attributes are the ranked list) andfinally the FCA technique is used for classification and analysis of this set.The main contributions of this paper are:

1. An interface and ontology based queries formulation to minimize the effect overlapping queries terms problem , selection of good queries impact the size and the pertinence of the ranked result list so developer spent little time to find the real relevant parts of each services .
2. IR techniques to locate parts of source code relevantto the computation of each servicein sourcebuilt following a functional approach where the all proposed approaches in the literature use object oriented approach.
3. An FCA approach to decompose, understand the software and to reduce the search space time (search space =the ranked list for each service) spent bydevelopers to find real relevant part for each service.

The rest of the paperis structured as follows: section 2 presents the related work, section3 give some background needed to understand the approach, section 4 details the proposed approach, section5 present the developed tool and the case study and finally section 6 conclude and give some perspectives.

2 Related Works

This paper addresses the problem of locating services (the first step to reengineer legacy software to SOA) as concepts (feature) location problem using IR techniques,so related works on migrating application towards SOA is given in 2.1 and related workson concepts location using IR is given in 2.2.

2.1Migrating Legacy Software towardsWeb Services

Many approaches had been done in the context of reengineering legacy to SOA paradigm from both industrial and academia. In[4] propose a functional approach starting from the identification of the functionalities to be reused and locate them in code source based on testing techniques, Sneed [14] propose a framework for wrapping legacy PL/I, COBOL, and C/C++ code to expose functions within the programs as web service, he argue that locating source code that implement the web services is challenging task and take business rule approach to do it where Chen et al [15] propose an ontology mapping based approach to locate services in source code, in [16] model driven approach to reengineer relational database towards web services. In [17] Canforaet al, an interaction driven approach istaken to migrate legacy interactive system towards web services he record the interaction between user an interface to generate an FSA (automate) to be replayed when service is invoked. In [23] ontology and FCA with information from static analysis of the software are used to migrate a legacy c software towards SOAparadigm.

2.2 Concept Location

Concept location identifies parts of a software system that implement a specific concept that originates from the problem or the solution domain. Concept location is a very common software engineering activity that directly supports software maintenance and evolution tasks such as incremental change and reverse engineering [18] now much software understanding approaches apply IR techniques to locate concepts in legacy software. Semantic of the software is hidden in the source code lexicon, looking to source code as text and applying IR tools reveal a lot of knowledge about the code source to support software evolution tasks. Concept location is started as concept assignment problem [19]. Wilde [20] develop the reconnaissance tool to identify concepts based on dynamic analysis of source code. In nowadays IR techniques are used to locate concepts [21, 22, 18, and 24] these approaches differ in the IR techniques used to index the software, the use ofinformation about source codeonly (static approach), dynamic information or both.

3 Background

This section provides brief background informationon Formal Concept Analysis (FCA).

3.1. Formal Concept Analysis

FCA was introduced by RudolfWill in 82s [5].FCA is technique for data analysis,knowledge representation an information management.FCA has been applied in several domains as linguistics, software engineering, psychology, Artificial Intelligence and information retrieval [7].FCA provides a mathematical notion of concept (set of object that shares the same properties) and concept hierarchies that is based on order and lattice theory. The starting point of FCA is the Formal Context,which describe a binary relation between a set of object and a set of attributes (i.e.properties) from a domain of interest .formally a formal context k=(G ,M,I) where:

1. G a set of (formal) objects
2. M a set of (formal) attributes
3. I: I$\sqsubseteq G \times M$.

The formal context is represented as a table where each object formsan arrow and attributes form columns. The cellule formed by intersection of row and column j contains×,if the objectof rowi have the attribute of column j.Given a set of objects O of some context (G, M, I), it is interesting to consider the set of attributes common to all those object .For any set O $\subseteq$G, the set O'is defined as O' = {m∈M|mIg for all g∈O}.Similarly we derive the set of all objects having all of attributes from a given context :For any set of attributes A $\subseteq$M , the set A' is defined as A'={g∈ G |mIg for all m ∈ A}.A pair (O,A) with O $\subseteq$G and A $\subseteq$M is a formal concept whenever we find O=A' and A=O' , here O is the extent and A is the intent of the formal concept .The set of all concept constitutes a lattice L when provided with the following specialisation order based on intent/extent inclusion .(O1,A1) <=L(O2,A2) ⇔O1$\subseteq$ O2 (i.e. A2$\sqsubseteq$ A1). For further understanding look at [13].

1.2.Vector Space Model

An algebraic Model used in information retrieval ,filtering and indexing to represent textual document as a vector of identifiers .In this model document (dj) and queries (q) are represented as a vector of terms: $d_j = (w_{1,j}, w_{2,j}, ..., w_{tj}), q = (w_{1,q}, w_{2,q}, ..., w_{t,q})$ where w is theweight attributed to term in the document ,absence of term in document is denoted by the weight 0.Tf-Idf is the most common approach used to compute this weight [29]. Cosine similarity is the common used similarity measure to calculate the relevance ranking of documents to a query [29], where the term specific weights in the document vectors are products of local and global parameters. The weight vector for document d is Vd=[w1,d,w2,d,…,wN,d]T , where$w_{t,d}=tf_{t,d} \cdot \frac{|D|}{|(d' \in D | t \in d')|}andtf_{t,d}$ is term frequency of term t in document d (a local parameter), $\log \frac{|D|}{|(d' \in D | t \in d')|}$ is inverse document frequency (a global parameter). | D | is the total number of documents in the document set, $|(d' \in D | t \in d')|$is the number of documents containing the term t. Using the cosine, the similarity between document dj and query q can be calculated as: $sim(dj,q) = \frac{dj.q}{\|dj\|\|q\|} = \frac{\sum_{i=1}^{N} wi,jwi,q}{\sqrt{\sum_{i=1}^{N} wi,j2}\sqrt{\sum_{i=1}^{N} wi,q2}}$

4 TheProposedApproach

The main steps of the approach are depicted in Fig.2 and the code source of Fig.1 is used as an example toexplain the concepts used in this approach.

```
#include<stdio.h>
#include<conio.h>
voidmyFunction();
intadd_two_numbers(int, int);
voidmain()
{clrscr();myFunction();
printf("\n\n%d",add(10,15));
getch();
}voidmyFunction()
{printf("This is inside function :D");}
intadd_two_numbers(intnumber_1, intnumber_2)
{returnnumber_1+number_2 ;}
```

Fig. 1 A sample source code

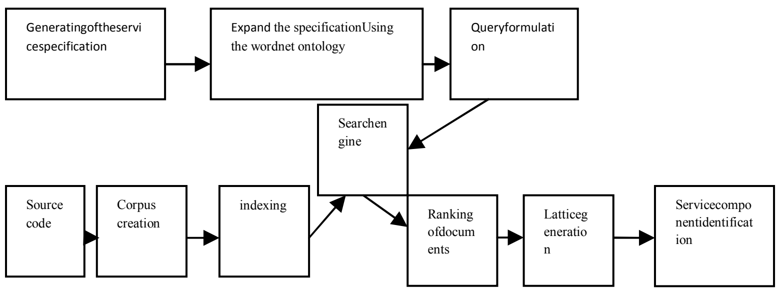

Fig.2 The proposed approach.

4.1 Generation of Service Specification

Legacy interactive software expose functionalities to end users via interfaces composed of inputsand outputsdata, terms are used to denote the data to be enteredin order to invoke the functionality or data that are visible to the user as a result of the

execution of thefunctionality and a name of the functionality.
We need to locate and extract each functionality and reuse it as web services .starting from this idea a specification can be generated for each future service based on the interface of each functionality, this specification is 3-uplet(inputs terms, outputs terms, name of the service), inputs terms denote the data to be entered to invoke the service, the output terms denote the result data if service is executed and name is the service name.

4.1.1 ExpandingtheSpecification

Legacy software is developed and maintained several times by different developpers, so different terms are used to denote the same concept WORDNET is used to calculate for each term the set of its synset (synonyms).

4.1.2 QueryFormulation

Each specification is mapped to query. This query is the union of the specification terms (and including terms result of the expansion in 4.2.)and the name of the service to be located (verb phrase).

4.1.3 Corpus Creation

Source code is analysed and modules (i.e. proceduresand functions)composing the legacy software are extracted to be used as documents(i.e. each module is a document),each future document will be composed only from comments and identifiers names (the rest is discarded) and a corpus is created as set of document each document represent a module composing the legacy software. As an example from the source code of fig.1 we construct a corpus composed from the three documents: main, myfunction, add.

4.2 Indexing

Before starting this task, each document is pre-processed so identifiers terms are split and stemmed then the corpus is indexed using the vector space model (other IR technique can be used) and a vector is generated for each document.As an example the terms extracted from function

```
intadd_two_numbers(intnumber_1, intnumber_2)
{ returnnumber_1+number_2 ; }
```

Of source code of fig.2 are: int, add, two, number (have 3 occurrences), return. A list of non informative word can be used to discard non informative terms as int, return and so on.

4.6 Rankingthe Documents

We calculate the similarity measure for query representing a service and the documents composing the corpus and the result is set of documents relevant to this query sorted according to their descending score of similarity. We take only first n document or we take the set of relevant document having score>a (a or n is fixed empirically).this process is repeated for each service query.

4.7 Lattice Generation

Results of step 5 are taken to build the context table in the following manner:
Each service is an object and each document figuring in the results is an attribute, a lattice of concepts is generated using the algorithm of[12].

4.8 ServiceLocation

The generated lattice give a good point to start filtering the real partcontributing in service computation,by starting examining shared modules. Sharedmodules are modules that participate in the computation of more than service, when this situation happen, itis easilyidentifiable in the lattice by looking for concepts (i.e. nodesin the lattice)having more than objects (i.e. service).
Developer start analysis of this concepts type to make decision about the shared modules if are:

1. Really shared modules:Partsparticipating really in the computation of different services.

2. Some are deceiving modules: Are not part of any services and this is fault of the IR techniques, or only participate in a set of services composing the concept (n services) with n<card (set objects composing the concepts).

3. All this modules do not participate in any services of this concept, this can happen when other services are not take into account when building the context table (we are not forced to take all functionalities but what we want to migrate only).

With this definition rules we reduce the time spent to decide which moduleshave to be examined first andbetter understanding can be gained.

As an example consider the following context, table 1, composed of three services s1, s2, s3 and four attributes p1...4 :

	P1	P2	P3	P4
S1	×	×		×
S2	×	×	×	
S3		×	×	

Table.1A context table example.

The lattice generated for this context is presented in Fig .3.

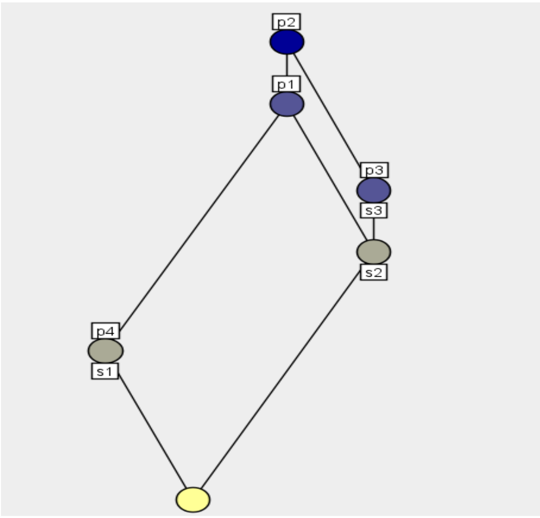

Fig.3: The lattice for the context table in table 1.

In the lattice, nodes (i.e. concepts) only shows the new objects (i.e. services) and attributes (i.e. modules) per concept, more formally a node is labelled with an object (O) if that node is the smallest concept with O in its extent and it is labelled with an attribute (I) if it is the largest concept with I in its intent.This approach is taken also in the case study, it facilitate the understanding of the lattice.From this latticewe can infer the following concepts:

C1{(s1),(p4,p1,p2)}
C2{(s2),(p1,p2,p3)}
C3{(s3),(p3,p2)}
C4{(),(p2)}
C5{(),(p1,p2)}

Concept c4,c5 are good starting point in reducing search space where p1 is shared between s3 and s, p2 is shared between s3 ,s2, s1 and we have to decide if it is.

5 Case Study

In order to evaluate the effectiveness of the proposed approach, atool is built. Code worker [25] andSRCML [26] are used to analysis and extract modules from the legacy c software,

The search engine is built using the LUCENE API [27],term sysnset are computed using wordnetAPI [30].

Using this tool we have evaluated this approach on Cairline reservation software download from http://www.planet-source-code.com, asite for sharing software source code.5 services are taken to be located in the legacy softawre,table.2 present each services with query used to locate it,the first 5 modules (top score) are taken from the ranked list:

ServiceName	**Query**	**The top 5 modules**
Addairline	Add airline +name+departuretime+arrivaltime+ capacity+aircraft type	Add_airline,option, edit_airline,option2, option1,
Delete anairline	Delete airline+name+capacity+type+departuretime+arrival time	Delete_airline, Add_airline,option, edit_airline,option2.
Searchanairline	Searchairline+name+destination	Option,option1,airlinef,view_airline,view_customer
Add a customer	Add customer+ name+ phone+nicnumber +passport +nationality	Add_customer,edit_customer,view_customer,customerf,delete_customer.
Delete a customer	Delete customer+name	Add_customer, edit_customer,view_customer,customerf,delete_customer

Table .2services to be located,the used query and the top 5 modules result of the query.

Fig.4 presents the generated lattice using TOSCANAJ tool [31].

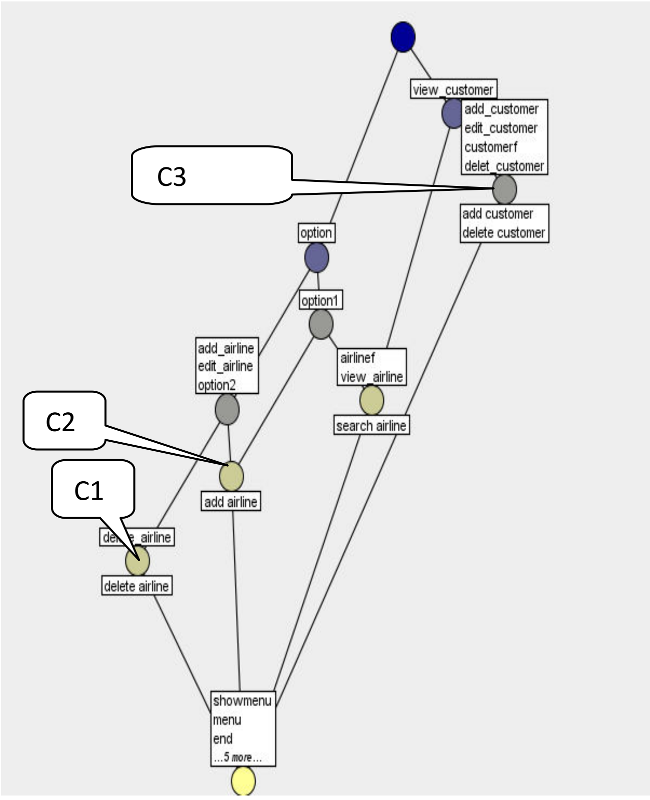

Fig. 4 The generated lattice for the airline reservation software

5.1 Analysis of the Lattice

We give some example on how we interpret the lattice to reduce search space time:
The concept c1 (i.e. delete an airline) have the module delete_airline as its proper module,where {Option, add_airline,edit_airline,option2} are also potentiel part of it but are shared with the service add an airlline (concept c2) ,option is shared with delete an airline,add an airline and search an airline the developer examine option module first,then pass to modules { add_airline,edit_airline,option2}, after examination we find that edit airline is not part of any of the two services (delete airline,add airline) and is discarded where add airline is part of add a airline service but option2 and option are part of delete airline.The concept c3 reveal that the service add customer and delete customer share the module set{add-customer,edit_customer,customerf} and the module customlerfis part of search airline ,add customer and delete customer services,we start with view customer is not part of any of the three.

6 Conclusion

In seventeenth Lehman introduced his law of software evolution [9]. Evolution is driven by two major forces: changes in the business domain of the application and changes (i.e. advances) in the software architectures, paradigm and technologies. SOA architecture and web services technologies tend to become the prevailing software engineering practice. In the SOA (i.e., Service Oriented Architecture) applications can be developed by composing existent services offered by different providers and the services web define the technology of this idea.

In this paper present an approach to locate services in legacy software is proposed.The identification and query formulation are interface and ontology (WORDNET) driven approach, IR techniques are used to locate relevant parts composing each services and FCA is used to reduce space search by detecting shared parts to give a good starting point for developer to filter the result list and reducing the time search space .A tool is built and a case study is conducted on real legacy software built using a functional approach, the approach shows that IR techniques can be used for concepts location in functional source code also .
In the future we plan to enhance the approach with informationgathered from the static and dynamic analysis of the software.

References:

[1] C. Abrams, R. W. Schulte. Service-Oriented Architecture Overview and Guide toSOA Research. Technical report G00154463, Gartner Research, January 2008.
[2]J.Chatarji: Introduction to service-oriented architecture (SOA) (2004)

[3]WILDE, N., GOMEZ, J.A., GUST, T. and STRASBURG, D. 1992. Locating User Functionality in Old Code.In*IEEE International Conference on Software Maintenance (ICSM'92)*, Orlando, FL, 200-205

[3] O'Brien, L., Smith, D., Lewis, G.: Supporting migration to services using softwarearchitecture reconstruction. In: Software Technology and Engineering Practice.(2005).

[4] Chen, F., Zhang, Z., Li, J., Kang, J., Yang, H.: Service identification via ontologymapping. In: Computer Software and Applications Conference. (2009).

[5]Sindhgatta, R., Ponnalagu, K.: Locating components realizing services in existingsystems. In: Services Computing. (2008)

[6] Li, S., Tahvildari, L.: E-BUS: a toolkit for extracting business services from javasoftware systems. (2008) 961{962

[7] Ilk, N., Zhao, J., Hofmann, P.: On reuse of source code components in modernizing enterprise systems. In: Advanced Management of Information for GlobalizedEnterprises. (2008)

[8] N. Anquetil and T. Lethbridge, “Extracting concepts fromfile names: A new file clustering criterion,” in *Proceedings ofthe* 20th *International Conference on Software Engineering*,

[9] Lehman, M.M.: On understanding laws, evolution and conservation in the large program life cycle. Systems and Software 1(3) (1980) 213–221.

[10] M. Eaddy, A. V. Aho, G. Antoniol, and Y.-G.Gueheneuc,“CERBERUS: Tracing requirements to source code usinginformation retrieval, dynamic analysis, and programanalysis,” in *Proceedings of the* 16th *InternationalConference on Program Comprehension (ICPC)*, IEEEComputer Society Press, June 2008.

[11] D. Poshyvanyk, Y.-G. Gu´eh´eneuc, A. Marcus, G. Antoniol,and V. Rajlich, “Feature location using probabilistic rankingof methods based on execution scenarios and informationretrieval,” *Transactions on Software Engineering (TSE)*,vol. 33, no. 6 June 2007.

[12]GANTER, B. and WILLE, R. 1996. *Formal Concept Analysis*. Springer-Verlag, Berlin, Heidelberg, New York

[13] Bernhard GanterGerdStummeRudolfWille Formal Concept Analysis Foundations and Applications : Lecture Notes in Artificial Intelligence ISSN 0302-9743 -2005

[14]. Sneed, H.M.: Integrating legacy software into a service oriented architecture. In: Proc.European Conf. Software Maintenance and Reengineering (CSMR), Los Alamitos, CA,USA, IEEE Computer Society Press (2006) 3–14

[15] Chen, F., Zhang, Z., Li, J., Kang, J., Yang, H.: Service identi_cation via ontologymapping. In: Computer Software and Applications Conference. (2009)

[16] Guzman,G.R., Polo, I., PiattiniM.:Anadm approach to reengineer relational database towards web services.

[17] Canfora, G., Fasolino, A.R., Frattolillo, G., Tramontana, P.:Migratinginteractivelegacy systems to web services. In: Proc. European Conf. Software Maintenance and Reengineering (CSMR), Washington, DC, USA, IEEE Computer Society Press (2006) 24–36

[18]Andrian Marcus, AndreySergeyev, VáclavRajlich, Jonathan I. Maletican Information Retrieval Approach toConcept Location in Source Codeproceedings of the 11th Working Conference on Reverse Engineering (WCRE’04)

[19] Biggerstaff, T. J., Mitbander, B. G., and Webster, D. E.,"Program Understanding and the Concept Assignment Problem",Comm. of the ACM, vol. 37(5), May 1994.

[20] Wilde, N. and Scully, M., "Software Reconnaissance:Mapping Program Features to Code", Journal of Software Maintenance and Evolution, vol. 7, 1995, pp. 49-62.

[21] Poshyvanyk, D., Marcus, A., and Dong, Y., "JIRiSS - an Eclipse plug-in for SourceCode Exploration", in Proc. of 14th IEEE International Conference on ProgramComprehension (ICPC'06), 2006.

[22] Poshyvanyk, D. and Marcus, D., "Combining Formal Concept Analysis withInformation Retrieval for Concept Location in Source Code", in Proc. of 15th IEEEInternational Conference on Program Comprehension (ICPC'07), 2007.

[23] MostefaiAekMalkiMimounBouchihadjeloul**,** An FCA and ontology based approach to reengineer Legacy software towards SOA architectures in international congress on Models, Optimization and Security of Systems (ICMOSS'2010) May 29-31, 2010.

[24] Poshyvanyk, D., Guéhéneuc, Y. G., Marcus, A., Antoniol, G., and Rajlich, V., "Feature Location using Probabilistic Ranking of Methods based on ExecutionScenarios and Information Retrieval", *IEEE Transactions on Software Engineering*,vol. 33, no. 6, June 2007.

[25]CodeWorker :codeworker.free.fr last visit 1/9/2011

[26]SRCML:www.sdml.info/projects/srcml last visit 1/9/2011

[27]Lucene : lucene.apache.org last visit 1/9/2011

[28]Miller, George A. WordNet: a lexical database for English. *Communications of the ACM* . (1995).

[29] Christopher D. Manning, PrabhakarRaghavan and HinrichSchütze Introduction to Information Retrieval..Cambridge University Press, 2008.
[30]javawordnetAPI :sourceforge.net/projects/jwordnet.

[31] TOSCANAJ:http://toscanaj.sourceforge.net/