Supplying Collaborative Source-code Retrieval Tools to Software Developers

Juan M. Fernández-Luna
Departamento de Ciencias de la Computación e Inteligencia Artificial, CITIC-UGR.
Universidad de Granada,
18071 Granada, Spain
jmfluna@decsai.ugr.es

Juan F. Huete
Departamento de Ciencias de la Computación e Inteligencia Artificial, CITIC-UGR.
Universidad de Granada,
18071 Granada, Spain
jhg@decsai.ugr.es

Julio C. Rodríguez-Cano
Centro de Desarrollo Territorial Holguín. Universidad de las Ciencias Informáticas, 80100
Holguín, Cuba
jcrcano@uci.cu

ABSTRACT

Collaborative information retrieval (CIR) and search-driven software development (SDD) are both new emerging research fields; the first one was born in response to the problem of satisfying shared information needs of groups of users that collaborate explicitly, and the second to explore source-code retrieval concept as an essential activity during software development process. Taking advantages of the recent contributions in CIR and SDD, in this paper we introduce a plug-in that can be added to the *NetBeans IDE* in order to enable remote teams of developers to use collaborative source-code retrieval tools. We also include in this work experimental results to confirm that CIR&SDD techniques give out better search results than individual strategies.

Categories and Subject Descriptors

H.5.3 [**Information Interfaces and presentation (e.g., HCI)**]: Group and Organization Interfaces; H.3.3 [**Information Storage and Retrieval**]: Search Process.

General Terms

Design, Human Factors.

Keywords

Collaborative Information Seeking and Retrieval, Search-driven Software Development, Multi-user Search Interface.

1. INTRODUCTION

"Collaboration" seems to be the buzzword this year, just like "knowledge management" was last year.
– David Coleman

In the last few years, Information Retrieval (IR) Systems have become critical tools for software developers. Today we can use vertical IR systems focused in integrated development environment (IDE) extensions for source-code retrieval as such *Strathcona* [5], *CodeConjurer* [6], and *CodeGenie* [1], but these only allow an individual interaction from the team developers's perspective.

One of the reasons that the existing IR systems do not adequately support collaboration is that there are not good models and methods that describe users' behavior during collaborative tasks. To address this issue, the community has adopted CIR as an emerging research field in charge to establish techniques to satisfy the shared information needs of group members, starting from the extension of the IR process with the knowledge about the queries, the context, and the explicit collaboration habits among group members. CIR community identifies four fundamental features in this multidisciplinary field that can enhance the value of collaborative search tools: user intent transition, awareness, division of labor, and sharing of knowledge [2].

In addition, SDD is a new research area motivated by the observation that software developers spend most of their time searching pertinent information that they need in order to solve their tasks at hand. We identified that SDD context was a very interesting field where collaborative IR features could be greatly exploited. For this reason we use the phrase collaborative SDD to refer to the application of different collaborative IR techniques in the SDD process [3].

It's known than some IDE incorporate tools with support for developer's collaboration practices, but without making emphasis in source-code retrieval. In this sense, the objective of this paper is to present the results of the comparison of traditional SDD and collaborative SDD. In both search scenarios, we use the *NetBeans IDE* plug-in *COSME* (COllaborative Search MEeting) with the appropriate configurations. *COSME* endows *NetBeans IDE* with traditional and collaborative source-code retrieval tools.

This paper is organized as follows: The first section presents a brief overview of related works and place our research in context. Then, we describe our software tool and method, explaining the different aspects of our experimental evaluation. Finally we discuss the results and present some conclusion remarks.

2. RELATED WORK

There is a small body of work that investigates methods to join collaborative information retrieval and search-driven software development. On the one hand, some researchers have identified different search scenarios where it is necessary to extend IR systems with collaborative capabilities. For example, in the Web context, *SearchTogether* [8] is a system which enables remote users to synchronously or asynchronously collaborate when searching the Web. It supports collaboration with several mechanisms of group awareness, division of labor, and persistence. On the other hand, the SDD community presents different prototypes and systems. For example, *Sourcerer* [1] is an infrastructure for large-scale indexing and analysis of open source code. *Sourcerer* crawls Internet looking for *Java* code from a variety of locations, such as open source repositories, public web sites, and version control systems.

CIR systems can be applied in several domains, such as travel planning, organizing social events, working on a homework assignment or medical environments, among many others. We identified software development as another possible application field where much evidence of collaboration among programmers on a development task can be found. For example, concurrent edition of models and processes require synchronous collaboration between architects and developers who can not be physically present at a common location [7].

However, current SDD systems do not have support for explicit collaboration among developers with shared technical information needs, which frequently look for additional documentation on the API (Application Programming Interface), read posts for people having the same problem, search the company's site for help with the API, or looking for source code examples where other people successfully used the API. Fortunately, in the last few years, some researchers have realized that collaboration is an important feature, which should be analyzed in detail in order to be integrated with operational IR systems, upgrading them to CIR systems.

As an approach to these situations, we propose in this work the *COSME* plug-in [4]. It makes the contribution in current SDD providing explicit support for teams of developers, enabling developers to collaborate on both the process and results of a search. *COSME* provides collaborative search functions for exploring and managing source-code repositories and documents about technical information in the software development context.

In order to support such CIR techniques, *COSME* provides some collaborative services in the context of SDD:

- The embedded chat tool enables direct communication among different developers.
- Relevant search results can be shared with the explicit recommender mechanisms.
- Another important feature is the automatic division of labor. By implementing an effective division of labor policy the search task can be split across team developers, thereby avoiding considerable duplication of effort.
- Through awareness mechanisms all developers are always informed about the team activities to save effort. Awareness is a valuable learning mechanism that help the less experienced developers to view the syntax used by their teammates, being an inspiration to reformulate their queries.
- All search results can be annotated, either for personal use, like a summary, or in the team context, for discussion threads and ratings.

3. THE COSME PLUG-IN

To improve software developers with shared technical information needs we implemented the *COSME* front-end as a *NetBeans IDE* plug-in. The principal technologies that we used to implement it include the *CIRLab* framework [2], *NetBeans IDE* platform, *Java* as programming language, and *AMENITIES* (A MEthodology for aNalysis and desIgn of cooperaTIve systEmS) as software engineering methodology. *COSME* is designed to enable either synchronous or asynchronous, but explicit remote collaboration among teams of developers with shared technical needs. In the following section we are going to outline *COSME*.

3.1 Current Features

Figure 1 is a screenshot showing various features of our *COSME* plug-in. We refer to the circled numbers in the following text.

1. Search Control Panel: It is integrated in turn for three collapsible panels; **(a)** configuration, where the developers can select the search options and engines to accomplish the search tasks; **(b)** filters show the user's interest field according to the collection contents; and **(c)** collection type permit to specify the type of search result's items.

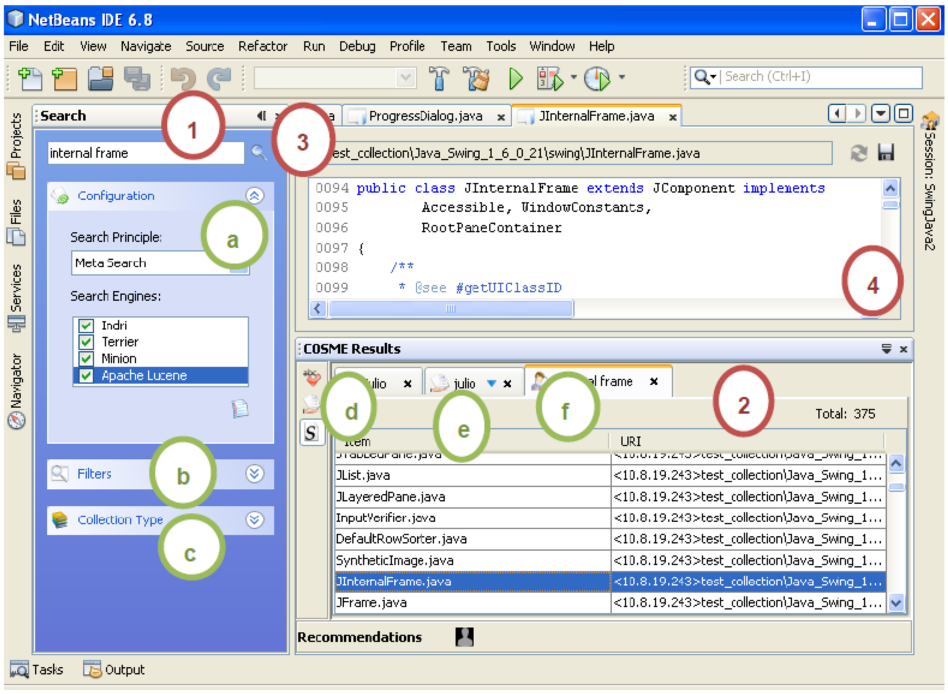
2. Search Results Window: The search results can be classified according to three different source-code localization: **(d)** results can be obtained as a consequence of division of labor techniques introduced by the collaborative search session (CoSS) chairman. A CoSS is a group of end-users working together to satisfy their shared information needs. One CoSS only can have one developer in the roll of chairman; **(e)** or by explicit recommendations accomplished for group members of their CoSS; **(f)** finally, search results also can be obtained by individual search.

3. Item Viewer: It shows full item content in different formats, e.g. pdf, plain text, and *Java* source-code files. All item formats are showed to the developers within the *NetBeans IDE*.

4. CoSS Portal: Developer can use the chat tool embedded in the CoSS Portal to negotiate the creation of a collaborative search session or to join at any active CoSS. For each CoSS, the chairman can to establish the integrity criteria, membership policy, and division of labor principles.

4. EXPERIMENTAL EVALUATION

In this section we are going to show how collaborative features applied to SDD improves the traditional operation without them. Then if we consider the null hypothesis (H_0) that $A_{TSDD} \geq A_{CSDD}$, our alternative hypothesis (H_1) is that the collaborative work should help to improve the retrieval performance in a SDD task: $A_{TSDD} < A_{CSDD}$, where TSDD stands for Traditional SDD and CSDD for Collaborative SDD. To evaluate our proposal we compare 10 group interactions in two different kinds of search scenarios (SS) on SDD, SS_{2k+1} and $SS_{2(k+1)}; k \in 0, \ldots, 9$. SS_{2k+1} represents a team of developers that use a conventional IR system, this means that developers do not have access to techniques of division of labor, sharing of knowledge, or awareness (traditional SDD – TSDD), while $S_{2(k+1)}$ represents a team of developers that uses a CIR system. Then, 5 teams worked in a TSDD context (those with odd subindexes) and the other 5 with CSDD (even subindexes). In both search scenarios, we used *COSME* with the appropriate configurations for both settings.



internal frame

Search Principle: Meta Search

Search Engines:

- ✓ Indri
- ✓ Terrier
- ✓ Minion
- ✓ Apache Lucene

est_collection\Java_Swing_1_6_0_21\swing\JInternalFrame.java

```
public class JInternalFrame extends JComponent implements
        Accessible, WindowConstants,
        RootPaneContainer
{
    /**
     * @see #getUIClassID
```

Total: 375

Item	URI
JList.java	<10.8.19.243>test_collection\Java_Swing_1...
JLayeredPane.java	<10.8.19.243>test_collection\Java_Swing_1...
InputVerifier.java	<10.8.19.243>test_collection\Java_Swing_1...
DefaultRowSorter.java	<10.8.19.243>test_collection\Java_Swing_1...
SyntheticImage.java	<10.8.19.243>test_collection\Java_Swing_1...
JInternalFrame.java	<10.8.19.243>test_collection\Java_Swing_1...
JFrame.java	<10.8.19.243>test_collection\Java_Swing_1...

Figure 1: Screenshot of *NetBeans IDE* with *COSME* plug-in installed

The search scenario was a common task proposed to a group of developers without *Java* background: select the most relevant classes to manage GUI (Graphical User Interface) components using different *Java* API with a total of 2420 files. Specifically, *Jidesoft* (634), *OpenSwing* (434)), *SwingX* (732)) and *Swing* (620). We have focussed on these API because they are directly related to the context of the experiment although they are not complete: we have only considered their most relevant API packages for the experiment.

For evaluation purposes, we created our own test collection: a group of 10 experts proposed a set of 100 topics strongly related to the objective of the experimentation, then their corresponding queries were submitted to each of the following search engines: *Lucene*, *Minion*, *Indri* and *Terrier*. A document pool was obtained by ranking fusion and later the experts, grouped in pairs, determined the relevant documents for each topic.

In collaborative SDD, it is very important to analyze the interaction among group members, therefore, unlike the evaluation of a traditional SDD system, we can not fix the queries. Then each participating group could freely formulate their queries to the search engine. In order to compare team results, the search engine identified the most similar queries formulated by the members of the teams with respect to those formulated by experts. If the system found enough similarity and if they occur in all the groups, then these queries are considered that deals with the same topic and selected for group comparison purposes. The similarity measure between queries is calculated by Equation 1. A user query (q_u) and an expert query (q_e) are considered to be the same if they are within a given similarity threshold. A new query $q_u{}'$ is obtained applying the Porter stemmer algorithm to q_u's terms, and analogously, we would obtain $q_e{}'$.

$$sim(q_u, q_e) = \frac{\mid q_u{}' \bigcap q_e{}' \mid}{\mid q_u{}' \bigcup q_e{}' \mid} = \lambda \qquad (1)$$

In Equation 1, λ is a value between 0 and 1. For this experiment we assumed that there exists an expert's relevance judgement to q_u only if $\exists\ \lambda \geq \frac{\frac{N}{2}+1}{N}$, where $N = \mid q_u{}' \bigcup q_e{}' \mid$, selecting the relevance judgements that correspond to λ_{max} for each q_e.

In order to measure the effectiveness of the described SS_{TSDD} and SS_{CSDD} scenarios, we considered as evaluation measures the metrics proposed by Pickens et al. in [9], i.e. selected precision (P_s, the fraction of documents judged relevant by the developer that were marked relevant in the ground truth), and selected recall (R_s) as their dependent measures. To summarize effectiveness in a single number we use F_{1s} measure.

According to the documents that each team selected for each common topic, F_{1s} measure was computed. In order to accomplish the statistical analysis of the results, we use the non parametric test of Wilcoxon (all against all). The Monte Carlo method was used and adjusted with the 99% trust intervals and 10000 signs. It was considered the existences of significance (Sig.) as appear in Table 1.

We could notice significative differences between TSDD and CSDD groups, considered two by two. As F_{1s} values for CSDD groups are better than those computed from TSDD groups for those cases, then we could conclude that when teams works supported by collaborative tools, they obtain better results. From Table 1, we could realize that apart from SS_5, each SS_{TSDD} has got at least one SS_{CSDD} with significant difference values of F_{1s}. With this results we accept H_1, because $A_{TSDD} < A_{CSDD}$.

	SS_1	SS_2	SS_3	SS_4	SS_5	SS_6	SS_7	SS_8	SS_9
					F_{1s}				
SS_2	$0,062$								
SS_3	$0,180$	$0,051$							
SS_4	$0,022^{\dagger}$	$0,212$	$0,038^{\dagger}$						
SS_5	$0,272$	$0,069$	$0,152$	$0,054$					
SS_6	$0,045^{\dagger}$	$0,201$	$0,080$	$0,290$	$0,056$				
SS_7	$0,215$	$0,031^{\dagger}$	$0,340$	0,090	$0,206$	$0,042^{\dagger}$			
SS_8	$0,053$	$0,131$	$0,061$	$0,190$	$0,072$	$0,158$	$0,070$		
SS_9	$0,243$	$0,072$	$0,201$	$0,029^{\dagger}$	$0,344$	$0,068$	$0,238$	$0,042^{\dagger}$	
SS_{10}	$0,065$	$0,098$	$0,041^{\dagger}$	$0,290$	$0,072$	$0,235$	$0,045^{\dagger}$	$0,132$	$0,058$

†: significant difference $(0,01 \leq Sig < 0,05)$
‡: highly significant difference $(Sig < 0,01)$

Table 1: Wilcoxon Test Results.

5. CONCLUSIONS AND FUTURE WORKS

Collaboration in SDD is just being recognized as an important research area. While in some cases collaborative SDD can be handled by conventional search engines, we need to understand how the collaborative nature of source-code retrieval affects the requirements on search algorithms. Research in this direction needs to adopt the theories and methodologies of SDD and CIR, and supplement them with new approach constructs as appropriate. In this work we present *COSME* as a collaborative SDD tool that helps team developers to find better sources than searching with traditional SDD strategies, as well as an experimental approach that confirms our hypotheses.

Our ongoing work focuses on the *COSME* back-end which poses fundamental research challenges as well as provides new opportunities to let group members collaborate in new ways:

(i) Profile Analysis. We aim to analyze the user-generated data using various techniques from the study of different collaborative virtual environments and recommender systems. With the results, our goal is to provide better personalized search results, support the users while searching and recommend users to relevant trustworthy collaborators.

(ii) P2P/hybrid-network Retrieval. Due to scalability and privacy issues we favor a distributed environment by means of a P2P (peer-to-peer) retrieval feature based on hybrid architecture to store the user-generated data and collections (*CASPER* – CollAborative Search in PEer-to-peer netwoRks). The main challenges in this respect are to ensure a reliable and efficient data analysis.

6. ACKNOWLEDGMENTS

This work has been partially supported by the Spanish research programme Consolider Ingenio 2010: MIPRCV (CSD2007-00018), the Spanish MICIN project TIN2008-06566-C04-01 and the Andalusian Consejería de Innovación, Ciencia y Empresa project TIC-04526. We also would like to thank Carmen Torres for support and discussions and for all of our experiment participants.

7. REFERENCES

[1] S. Bajracharya, J. Ossher, and C. Lopes. Sourcerer: An internet-scale software repository. In *SUITE '09: Proceedings of the 2009 ICSE Workshop on Search-Driven Development-Users, Infrastructure, Tools and Evaluation*, pages 1–4, Washington, DC, USA, 2009. IEEE Computer Society.

[2] J. M. Fernández-Luna, J. F. Huete, R. Pérez-Vázquez, and J. C. Rodríguez-Cano. Cirlab: A groupware framework for collaborative information retrieval research. *Information Processing and Management*, 44(1):256–273, 2009.

[3] J. M. Fernández-Luna, J. F. Huete, R. Pérez-Vázquez, and J. C. Rodríguez-Cano. Improving search–driven development with collaborative information retrieval techniques. In *HCIR '09: IIIrd Workshop on Human–Computer Interaction and Information Retrieval*, Washington DC, USA, 2009.

[4] J. M. Fernández-Luna, J. F. Huete, R. Pérez-Vázquez, and J. C. Rodríguez-Cano. Cosme: A netbeans ide plugin as a team–centric alternative for search driven software development. In *Group 2010: Ist Workshop on Collaborative Information Seeking*, Florida, USA, 2010.

[5] R. Holmes. Do developers search for source code examples using multiple facts? In *SUITE 2009: First International Workshop on Search-Driven Development Users, Infrastructure, Tools and Evaluation*, Vancouver, Canada, 2009.

[6] W. Janjic. Lowering the barrier to reuse through test-driven search. In *SUITE 2009: First International Workshop on Search-Driven Development Users, Infrastructure, Tools and Evaluation*, Vancouver, Canada, 2009.

[7] M. Jiménez, M. Piattini, and A. Vizcaíno. Challenges and improvements in distributed software development: A systematic review. 2009.

[8] M. R. Morris and E. Horvitz. Searchtogether: an interface for collaborative web search. In *UIST '07: Proceedings of the 20th annual ACM symposium on User interface software and technology*, pages 3–12, New York, NY, USA, 2007. ACM.

[9] J. Pickens, G. Golovchinsky, C. Shah, P. Qvarfordt, and M. Back. Algorithmic mediation for collaborative exploratory search. In *SIGIR '08: Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 315–322, New York, NY, USA, 2008. ACM.