INTERFACES CIENTÍFICAS

EXATAS E TECNOLÓGICAS

ISSN IMPRESSO - 2359-4934
ISSN ELETRÔNICO - 2359-4942
DOI-10.17564/2359-4942.2018v3n1p9-18

AN INFORMATION RETRIEVAL APPROACH FOR ASSISTING USERS IN SOFTWARE ENGINEERING PROCESSES

UMA ABORDAGEM DE RECUPERAÇÃO DE INFORMAÇÃO PARA APOIAR USUÁRIOS EM PROCESSOS DE ENGENHARIA DE SOFTWARE

UN ENFOQUE DE RECUPERACIÓN DE INFORMACIÓN PARA AYUDAR A LOS USUARIOS EN LOS PROCESOS DE INGENIERÍA DE SOFTWARE

Guillermo Rodriguez[1]

ABSTRACT

The documents written in natural language constitute a major part of the artifacts produced during the software engineering life cycle. There is a growing interest in creating tools that can assist users in all phases of the software life cycle. The assistance requires techniques that go beyond traditional static and dynamic analysis. An example of such a technique is the application of information retrieval (IR), which exploits information found in documents of a software engineering process. The increased availability of data created as part of the software development process allows managers to apply novel analysis techniques on the data and use the results to guide the project's stakeholders. These data are then used to predict defects, gather insight into a project's life-cycle, and other tasks. This work proposes an IR approach to assist users in software engineering processes according their profile. The approach consists in recommending them related documents to a retrieved one in order to users understand and follow the process in a correct way. Furthermore, the assistance concentrates on legacy systems in which engineers must acquire knowledge generated by others. Implementation of the approach and an overview of evaluation are also summarized.

KEY WORDS

Information Retrieval, Software Engineering, CMMI, Tool Support, Software Processes.

RESUMO

Os documentos escritos em linguagem natural constituem uma parte importante dos artefatos produzidos durante o ciclo de vida da engenharia de software. Há um interesse crescente em criar ferramentas que possam ajudar os usuários em todas as fases do ciclo de vida do software. A assistência requer técnicas que vão além da análise tradicional estática e dinâmica. Um exemplo de tal técnica é a aplicação de recuperação de informações (RI), que explora informações encontradas em documentos de um processo de engenharia de software. A maior disponibilidade de dados criados como parte do processo de desenvolvimento de software permite que os gerentes apliquem técnicas de análise inovadoras nos dados e usem os resultados para orientar as partes interessadas do projeto. Esses dados são usados para prever defeitos, coletar informações sobre o ciclo de vida de um projeto e outras tarefas. Este trabalho propõe uma abordagem de RI para auxiliar os usuários em processos de engenharia de software de acordo com o seu perfil. A abordagem consiste em lhes recomendar os documentos relacionados para que sejam recuperados, a fim de que os usuários compreendam e sigam o processo de maneira correta. Além disso, a assistência concentra-se em sistemas legados nos quais os engenheiros devem adquirir conhecimento gerado por outros. O desenvolvimento da abordagem e uma visão geral da avaliação são apresentados.

PALAVRAS-CHAVE

Recuperação de informação, Engenharia de Software, CMMI, Ferramentas de suporte, Processo de Software

RESUMEN

Los documentos escritos en lenguaje natural constituyen una parte fundamental de los artefactos producidos durante el ciclo de ciclo de vida del desarrollo de software. Hay un creciente interés en las herramientas de creación que pueden ayudar a los usuarios en todas las fases del ciclo de vida del software. La asistencia requiere técnicas que van más allá del análisis estático y dinámico. Por ejemplo, la aplicación de recuperación de información (RI) explota información encontrada en los documentos del proceso de desarrollo de software. La creciente disponibilidad de datos creados a partir del proceso de desarrollo de software permite a los líderes de proyectos aplicar estrategias de análisis sobre los datos como parte del proceso y utilizar los resultados para guiar a los grupos de interesados. Estos datos se utilizan para evitar defectos, recolectar información sobre el ciclo de vida de un proyecto, y otras tareas. En este trabajo se propone un enfoque de RI para asistir a los usuarios en el proceso de desarrollo de software según su perfil. El enfoque consiste en recomendar los documentos relacionados a un usuario para que éstos entiendan y sigan el proceso en una dirección correcta. Además, la asistencia está pensada en contextos de sistemas heredados en los que los desarrolladores deben adquirir el conocimiento generado por terceros. Como conclusión, se muestran la implementación del enfoque y una descripción de la evaluación.

PALABRAS CLAVE

Recuperación de información, Ingeniería de software, CMMI, Herramientas de soporte, Proceso de software

1 INTRODUCTION

In recent years, traceability has been more and more universally accepted as being a key factor for the success of software development projects (SCHWARZ; EBERT; WINTER, 2010). Particularly, traceability has focused on requirements as they constitute the first step in a software engineering project. Performing requirement traceability is twofold: (a) ensuring that a new system does indeed satisfy all its specified requirements, and (b) performing impact analysis on proposed changes (HAYES; DEKHTYAR; OSBORNE, 2003). In addition, managing traceability and context-aware is useful to understand legacy systems in order to evolve them. Documents are crucial artifacts in software engineering, especially during software maintenance or reverse engineering, semantic information conveyed in these documents can provide important knowledge for the software engineer (WITTE; LI; ZHANG; RILLING, 2008).

Managing their traceability allows stakeholders to be aware of system evolution. However, the variety of different approaches and technologies impedes the application of traceability techniques in practice. Furthermore, the lack of integration between tools adopted in the development processes is one of the main causes of ineffective management, where traceability relationships are still manually generated and maintained (AMALFITANO; DE SIMONE; FASOLINO; SCALA, 2017).

To cope with this issue, this work presents a view on traceability, pertaining to the whole software development process. Software organizations have to deal with integration issues to enable communication between tools and to properly support the software development process. For this reason, it is crucial to address integration not only at the syntactic level, but also at the semantic one (FONSECA; PIERINI BARCELLOS; DE ALMEIDA FALBO, 2017). Based on graph technology, this work derives a seamless approach which combines all activities related to software development process. In the course of this process, several artifacts are produced, ranging from collections of requirement statements over architecture and design models to source code and test cases. All these artifacts are strongly coupled. They may build on each other, or their individual elements may contain references to other elements in other software artifacts.

In this work, jDocRecommender is presented. This tool allows users involved in a software engineering process to access to documents from the Process Assets Library (PAL) in a personalized way. Accordingly, documents are retrieved by users taking into account their pro le into the software process and their background in order to be able to fully understand a retrieved document.

The remainder of the paper is organized as follows: Section 2 describes the main motivations that led to the emergence of jDocRecommender. Section 3 gives an overview of document retrieval. Section 4 describes user profiling in software engineering. Section 5 describes the implementation of jDocRecommender. Section 6 analyzes the evaluation methodology. Finally, Section 7 concludes this work.

2 MOTIVATION

The focus of this work was set in the context of the Software Engineering (SE) course of the Systems Engineering BSc program at the Faculty of Exact Sciences (Department of Computer Science – UNICEN, Argentine). In the current curriculum of System Engineering studies at UNICEN University, the SE course is divided into two parts. The first is compulsory during the first semester and the second part is elective during the second semester. Students have completed the following courses: Operating Systems, Databases, Methodologies of Software Development, Network and Communication Services, Object-Oriented Programming and Design of Software Systems.

The SE course attempts to teach students to recognize an engineering task and respond to it with an appropriate and standard technique to produce high--quality software artifacts. Thus, the course is intended to simulate a real software project and consider students like an organization who must carry out its development applying the concepts acquired in the previous courses.

During the first semester students are given the relevant topics of SE in order to prepare the process whi-

ch will guide the capstone project by using the Capability Maturity Model Integration (CMMI). This project is inherited from previous years. The aim is that project evolves year after year. Also, professors complement the course with anecdotes and previous experiences in companies in order to emphasize pedagogical techniques. To involve students in the teaching process, they are divided into groups and have to give a class explaining an assigned process area. Also, they have to design the process for this area with all the needed artifacts. They are assisted and guided by the professors to prepare the class to their own partners. Didacticism, oral presentation and quality of examples are highly taken into account to evaluate the group's presentations.

Professors base on Jigsaw technique in which students teach knowledge to others (ANDREAS; TSIATSOS; TERZIDOU; POMPORTSIS, 2010). Simultaneously, students are given a software development environment. It consists of a set of tools applied to carry out the process practices and activities according to the defined process. The tools were selected in order to be able to work in a distributed environment because students are physically distributed and had their times, obligations and extracurricular work.

In the second semester, the course is oriented to a software organization in which the processes designed in the previous term are executed. To perform that, the organization has to apply all the learned concepts to the development of a real and supervised capstone project. When students receive the project, they also receive the PAL full of documents completed during previous years. The main problem is students do not have the background enough to acquire all the knowledge in a short period of time. The professor plays the role of customer and end user. He encourages following the defined process and using the tools in order to maintain traceability of requirements.

To sum up, this work presents an IR-based approach to deal with the problem of legacy project. This approach is not only helpful from an educational perspective, but also jDocRecommender would be very interesting in software organizations in order to assist their members.

3 DOCUMENT RETRIEVAL APPROACH

Information contained in software documents is important for a multitude of software engineering tasks, particularly in concept location and traceability across different software artifacts (WITTE; LI; ZHANG; RILLING, 2008). From a maintainer's perspective, software documentation contains valuable information of both functional and non-functional requirements, as well as information related to the application domain. This knowledge often is di cult or impossible to extract only from source code (LINDVALL; SANDAHL, 1998).

It is a well-known fact that even in organizations and projects with mature software development processes, software artifacts created as part of these processes end up to be disconnected from each other (ANTONIOL; CANFORA; CASAZZA; DE LUCIA, 2000). As a result, maintainers have to spend a large amount of time on synthesizing and integrating information from various information sources in order to re-establish the traceability links among these artifacts.

One way to link software documents in order to achieve traceability is considering their content. In information retrieval, terms of a document d i can be used as content identifiers to represent its main concepts. This set of terms $t_{i1}, t_{i2}, \ldots, t_{im}$ is crucial to distinguish a document from others. A term-weight w_{ij}, ranging between 0 and 1, represents the degree of importance of terms t_{ij} in document d_i . A term weight close to 1 indicates that the term is important to the document.

$$\frac{tf * idf}{\sqrt{\sum_{i=1}^{m}(tf_i * idf_i)^2}}$$

A short-term vector represents a short document, whereas a long-term vector represents a long document. When a large number of terms are used for document representation, it is very likely to match terms between queries and documents. As a consequence, long documents have high chances of being retrieved

than short ones. Nevertheless, all relevant documents should be treated as equally important in the retrieval process. Thus, a normalization factor is incorporated into the term-weighting formula to equalize the length of the document vectors (EQ.1).

The most effective method to calculate the weight of the terms of a document is to multiply its within-document term frequency (tf) by its inverse document frequency (idf), and then normalize the product using the cosine measure. A vector of m terms $<t_{i1}, t_{i2}, \dots, t_{im}>$ is calculated using tf_i and idf_i ($1 \leq i \leq m$).

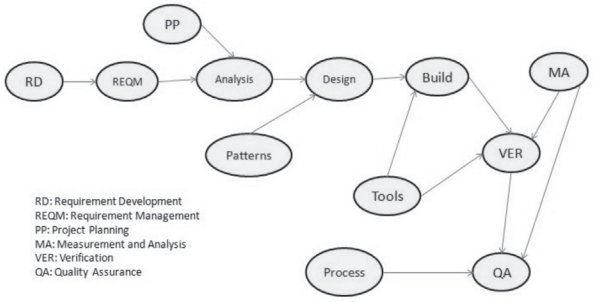
In order to calculate the similarity between a document d_i with others, the weighting vector is used. Accordingly, d_i corresponds to the vector $< w_{i1}, w_{i2}, \dots, w_{im} >$, whereas each w_{ij} indicates the weight or existence of term t_{ij} in document d_i. A set of n documents can be mapped into a $n \times n$ document similarity matrix by a selected mapping process, and each element s_{ij} of this matrix indicates the similarity of documents d_i and d_j.

The cosine measure is a common metric to calculate the similarity between two documents d_i and d_j. This measure defines the angle between tow vectors in a $n \times n$ matrix to quantify the similarity. For example, given two documents d_i and d_j, formula 2 describes the metric:

$$similarity(d_i, d_j) = \sum_{i=1}^{m} w_{ik} * w_{jk}$$

where w_{ik} and w_{jk} denote the term weights of t_{ik} and t_{jk}, respectively.

Figure 1 – Process Management Graph



Fonte: Prepared by the author

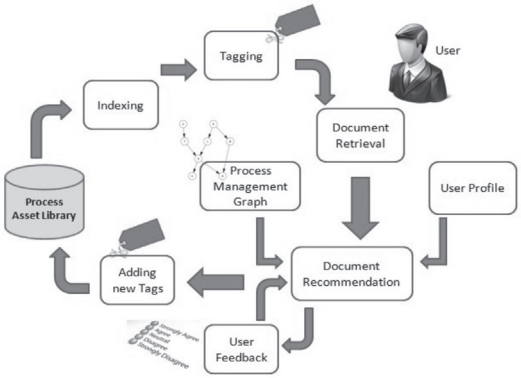
4 JDOCRECOMMENDER APPROACH

jDocRecommender is a tool that aims to assist users in a software engineering process. There are some traceability approaches based on graph dependency in Configuration Management (SCHWARZ; EBERT; WINTER, 2010), but there is not an approach to deal with process management dependencies. Thus, it is interesting to project manager to have a tool to assist project members in information accessing according to their pro le in the organization. As a first approach, this work proposes a graph-dependency view to link software engineering areas, artifacts and practices according to a process definition.

Fig. 1 shows a process management dependency graph in which most important practices and artifacts are linked. This graph was based on a process defined by students in the Software Engineering course. This work focuses on assisting students in accessing documents to fully understand the context of the project taking into account their role, profile, background and related knowledge needed to follow a particular document. The graph nodes are CMMI areas such as RD (Requirement Development), REQM (Requirement Management), PP (Project Planning), MA (Measurement and Analysis), VER (Verification), QA (Quality Assurance), software artifacts such as Process, Tools, Patterns and software practices like Analysis, Design and Build. Fig. 2 describes the general approach of jDocRecommender.

The approach consists in indexing documents of the PAL using Lucene2. Then, users search for documents using keywords. On documents retrieved, users are able to tag them according to the process management graph. These tags are used to evaluate documents and recommend them to users considering the user pro le and the graph. Once users are given the appropriated documents to understand a retrieval one, users are able to add new tags from the existing set based on the graph. Also, they provide an explicit feedback for the recommendation which will be used to train the tool. Documents tagged by users return to the PAL for being used in future retrievals.

Figure 2 – jDocRecommender approach



Fonte: Prepared by the author

4.1 USER PROFILING IN SOFTWARE ENGINEERING

A user pro le is a representation of information about an individual user that is essential for the application we are considering (SCHIAFFINO; AMANDI, 2009). User pro ling implies inferring unobservable information about users from observable information about them, that is, their actions or utterances. User pro les or user models are vital in many areas in which it is essential to obtain knowledge about users of software applications. Examples of these areas are intelligent agents, adaptive systems, intelligent tutoring systems, recommender systems, intelligent e-commerce applications, and knowledge management systems.

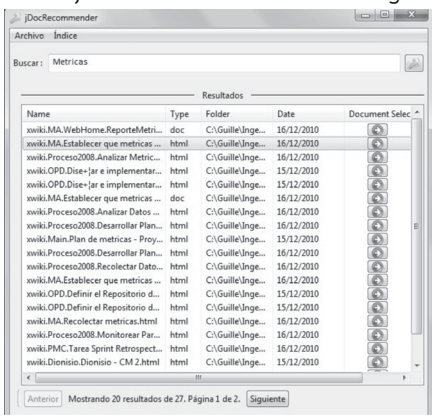
2 http://lucene.apache.org/java/docs/index.html

This work focuses on background and skills of users. Taking into account user pro les are crucial to provide personalized assistance to users. According to jDocRecommender approach, the following pro les were defined: QA Engineer, Project Leader, Software architect, Requirement Analyst, Developer, Manager, and Tester. Thus, a set of tags was defined for each profile in order to represent their background:

- QA Engineer: QA - MA - Process.
- Project Leader: REQM - PP – MA.
- Software architect: Design – Patterns.
- Requirement Analyst: RD - REQM – Analysis.
- Developer: Build - VER – Tools.
- Manager: MA - Process – PP.
- Tester: Tools - VER – Build.

Users provide their role in the software process and jDocRecommender defines their profile according to the tags. Also, user feedback is another key source of information to consider for improving future recommendations. This feedback is explicit, when users evaluate the assistance through a user interface provided for that purpose. In the explicit feedback user is required to evaluate the assistance according to a qualitative scale based on a 6-point Likert's scale (LIKERT, 1932): strongly agree, very agree, somewhat agree, somewhat disagree, very disagree and strongly disagree.

Figure 3 – jDocRecommender searching interface



Fonte: Prepared by the author

4.2 IMPLEMENTATION

To implement the approach a document searcher was implemented. The searcher allows users to index documents with different formats such as Portable Document Format (PDF), Microsoft Office (DOC, DOCX, XSL, XSLX, VSD, PPT), plain text (TXT), HyperText Markup Language (HTML) and Extensible Markup Language (XML). To do that, the following libraries were used: POI (Microsoft office documents), PDF-BOX (pdf documents), NekoHTML (HTML les), and SAX (XML les). On the other hand, the searcher permits to do queries on indexed documents using a single word o group of words present in the name or content le. In order to carry out the indexing and search engine, Lucene by Apache foundation was used.

The integrated development environment was Eclipse and programming language was Java. User interface is through a desktop interface application using Jigloo Eclipse plug-in3.

Fig. 3 shows the first part of jDocRecommender interface. In this part, an index can be created. On that index, all documents from the PAL are loaded. Also, new les are able to be added to that index. Documents are listed according to user input. The le name, type, folder and date are shown. When user is interesting in a document of the list, he/she can select it and the tool will retrieve the similar ones to it using More Like This. Those documents will be used in the next stage of recommendation.

4.3 DROOLS

In order to implement the process management graph, a rule engine was used. A rule engine helps engineers resolve (or at least reduce) the issues and difficulties inherent in the development and maintenance of an application's business logic. Engineers can think of a rule engine as a framework for implementing complex business logic. Most rule engines let engineers use declarative programming to express the consequences that are valid given some

Figure 4 – Drools example

```
rule "QA"
dialect "java"
when
t : Tag(tag == "QA", tag : tag)
then
        t.addTag("TEST");
        t.addTag("MA");
        t.addTag("PROCESS");
        update(t);
end
```

Fonte: Prepared by the author

information or knowledge. Engineers can concentrate on facts that are known to be true and their associated outcomes that is, on an application's business logic. Several rule engines are available, including commercial and open source choices. In this work, Drools was used as part of the business logic layer in the Java application.

Drools3 is an open source rule engine, written in Java language that uses the Rete algorithm to evaluate the rules that engineers write. Drools lets engineers express their business logic rules in a declarative way. Engineers can write rules using a non-XML native language that is quite easy to learn and understand. And they can embed Java code directly in a rules le, which makes the experience of learning Drools even more attractive (CRASSO; ZUNINO; MORENO; CAMPO, 2009).

Fig. 4 shows an example of a rule according to a document tag. This rule checks the corresponding tags to be considered in order to recommend the suitable documents for a user. The graph shown in Fig. 1 was implemented using Drools and Fig. 4 indicates that the tag QA is reachable if and only

3 http://www.cloudgarden.com/jigloo/

if the tags TEST (VER), MA and PROCESS were visited by the user.

4.4 METHODOLOGY OF EVALUATION

As a methodology to evaluate the approach, the number of correct assistance actions was considered. The correctness is determined by the user through explicit feedback. To do this, we defined a precision metric that measures jDocRecommender to accurately assist a user. As a 6-point Likert scale was defined to show the possible choices for users to select the feedback, a weighting vector v was built. Be m = 6 the number of possible choices, a probability of success was assigned to each option in every assistance:

v_1= Strongly agree (1.0), v_2= Agree (0.8), v_3=Somewhat Agree (0.6). v_4 = Somewhat Disagree (0.4), v_5= Disagree (0.2), v_6=Strongly Disagree (0.0).

Be k the selection index of the feedback list after each assistance and n the assistance set size, the precision is defined as follows:

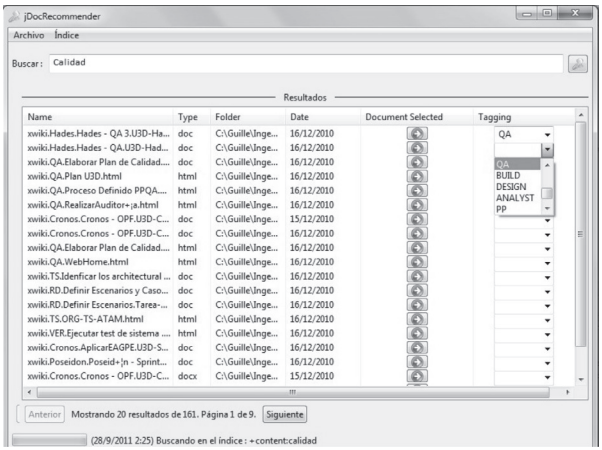
$$Precision = \frac{\sum_{i=1}^{n} v_{ik}}{n}$$

Figure 5 illustrates and example of searching documents by introducing a keyword. The table lists the documents retrieved through the keyword. Last column of the table shows how expert users tag documents. Fig. 6 depicts the recommendation interface. User selects a document in the previous interface. He/she wants the documents needed to understand the selected one. Thus, the user is required to complete the user name and select the role in the organization. Also, the user is able to add more tags to the selected document. When he/she presses the search button, a set of documents are recommended considering the user pro le and process management graph seen in Fig. 1. Finally, the user gives an explicit feedback in order to evaluate the recommendation.

5 CONCLUSIONS

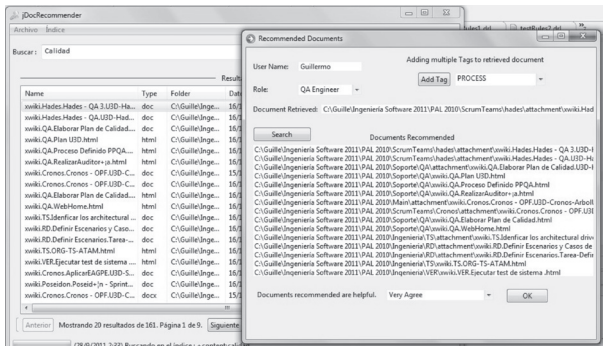
This work demonstrated the integration of information retrieval techniques for assisting users in following software engineering processes. This paper introduced an approach to maintain traceability by tagging software engineering documents in order to assist users in the retrieval process.

Figure 5 – Searching and indexing in jDocRecommender



Fonte: Prepared by the author

Figure 6 – Recommending and validating documents



Fonte: Prepared by the author

Additionally, we presented an approach for document retrieval, tagging and personalized recommendation according to a process management graph and user profile in a defined software engineering process. A user involved in a software engineering project can access to information according to their knowledge, instead of surfing on a huge amount of information that can discourage him/her. It is worth noting that jDocRecommender is useful for students who are involved in a capstone project, and for users involved in a company in which they can be assisted in order to access to the know-how. In summary, the approach recommends documents to users according to their context and profile.

Then, user feedback based on a qualitative explicit feedback was collected. As future work, this work will concentrate on taking into account the user feedback to adjust the assistance model and offer more precise recommendations. Also, the objective is to validate the tool with students of the SE course.

REFERENCES

AMALFITANO, Domenico *et al*. Improving traceability management through tool integration: an experience in the automotive domain. In: **Proceedings of the 2017 International Conference on Software and System Process**. ACM, New York, NY, USA, 2017.
ANDREAS, Konstantinidis *et al*. **Fostering collaborative learning in second life:** Metaphors and affordances. Computers and Education, 2010.

ANTONIOL, Giuliano *et al*. Information retrieval models for recovering traceability links between code and documentation. In: **Proceedings of International Conference on Software Maintenance**, San Jose, CA, USA, 2000.

CRASSO, Marco *et al*. **JEETuningExpert:** A software assistant for improving java enterprise edition application performance. Expert Systems with Applications, 2009.

FONSECA, Vinícius Soares; PIERINI BARCELLOS, Monalessa; DE ALMEIDA FALBO, Ricardo. An ontology-based approach for integrating tools supporting the software measurement process. **Science of Computer Programming**, 2017.

HAYES, Jane Huffman; DEKHTYAR, Alex; OSBORNE, James. Improving requirements tracing via information retrieval. **IEEE Computer Society**, 2003.

LINDVALL, Mikael; SANDAHL, Kristian. How well do experienced software developers predict software change? **Systems and Software**, 1998.

LIKERT, Rensis. A technique for the measurement of attitudes. **Arch Psychol**, 22, 1932.

SCHIAFFINO, Silvia; AMANDI, Analia. Intelligent user profiling. In: BRAMER, Max (ed.). **Artificial intelligence, chapter Intelligent user profiling**. Springer-Verlag, Berlin, Heidelberg, 2009.

SCHWARZ, Hannes; EBERT, Jürgen; WINTER, Andreas. **Graph-based traceability:** a comprehensive approach. Software and Systems Modeling, 2010.

WITTE, René *et al*. **Text mining and software engineering:** an integrated source code and document analysis approach. IET Software, 2008.

Recebido em: 8 de Fevereiro 2018
Avaliado em: 10 de Maio 2018
Aceito em: 25 de Maio 2018

1 Investigador de la Faculdade de Ciencias Exactas de Universidad Nacional del Centro de la Provincia de Buenos Aires (UNCPBA-Argentina), obtuve el título de Ingeniero de Sistemas en 2010 y el título de Doctor en Ciencias de la Computación (UNCPBA) en 2014.E-Mail: exarodriguez@gmail.com