Pre-Conceptual Schemas: Ten Years of Lessons Learned about Software Engineering Teaching

Juan Sebastián Zapata-Tamayo
Universidad Nacional de Colombia, Colombia
jzapatat@unal.edu.co

Carlos Mario Zapata-Jaramillo
Universidad Nacional de Colombia, Colombia
cmzapata@unal.edu.co

ABSTRACT

Pre-conceptual schemas are models for representing knowledge about any domain. Pre-conceptual schemas have been used for teaching software engineering for ten years in the National University of Colombia, since they are the basis for the so-called UNC-Method, a software development method created by researchers of such University. Several generations of software engineers have been taught with such models and, consequently, in this paper we survey some of them for gathering the lessons learned from such experience. Results are clear about the advantages and usage of pre-conceptual schemas for teaching software engineering. We also summarize such results and provide feedback for future development of pre-conceptual schemas.

INTRODUCTION

Software development is inherently complex (Booch *et al.*, 2007). Therefore, software engineering professors are searching for ways to ease the complexity of software development applications. Knowledge representation is one of the alternatives used for reaching such a goal. In fact, several work products have been developed over time in order to simplify such complexity, and UML (Unified Modeling Language) has some of them for representing the structure, dynamism, and behavior of a software application (Fowler, 2004). However, conceptual schemas—like those included in UML—are difficult to understand for stakeholders—the source of information for developing software applications (Haumer *et al,* 2000). For this reason, some researchers belonging to the Computing Languages Research Group of the National University of Colombia developed the so-called pre-conceptual schemas as an intermediate language for automatically developing UML work products (Zapata *et al,* 2006).

Zapata (2012) incorporates pre-conceptual schemas to the UNC-Method—the software development method used in the National University of Colombia for teaching software development—in 2007, as a way to deal with the communication problems the students face when gathering requirements from stakeholders. Since then, several generations of software engineers recognize pre-conceptual schemas being good structures for representing knowledge related to any domain. Such opinions lead us to survey some of the current users of pre-conceptual schemas in order to capture some lessons learned throughout ten years about using pre-conceptual schemas for teaching software engineering. Consequently, in this paper we summarize the results of such a survey and discuss all the feedback provided for improving the use of pre-conceptual schemas.

The structure of this paper is the following: first, we present some theoretical framework about pre-conceptual schemas; after that, we propose the structure of the survey and we summarize the main results; finally, we discuss conclusions and state future work.

THEORETICAL FRAMEWORK

PRE-CONCEPTUAL SCHEMAS

Zapata *et al.* (2006) introduce the term pre-conceptual schema as a way to differentiate well-known conceptual schemas—*e.g.,* the *de facto* UML standard diagrams like class, use case, and sequence diagrams—from another kind of knowledge representations closer to the stakeholder discourse. In fact, even though pre-conceptual schemas are conceptual models like the aforementioned ones, their nature is very intuitive and so their users have the possibility to previously understand the concepts and relationships belonging to any domain. Consequently, pre-conceptual schemas can be used for dealing with the problems revealed by Haumer *et al.* (2000): stakeholders misunderstand the well-known diagrams analysts use in the software development lifecycle due to their complex syntax. Basic elements of the pre-conceptual schemas are depicted in Exhibit 1.

In the 2007 version of the pre-conceptual schemas, the basic symbols were: concepts (nouns and noun phrases), structural relationships (verbs "is" and "has"), dynamic relationships (activity verbs), connections, implications, conditionals, and references.

EXHIBIT 1.
BASIC SYMBOLS OF PRE-CONCEPTUAL SCHEMAS
(SOURCE: ADAPTED FROM ZAPATA, 2012)

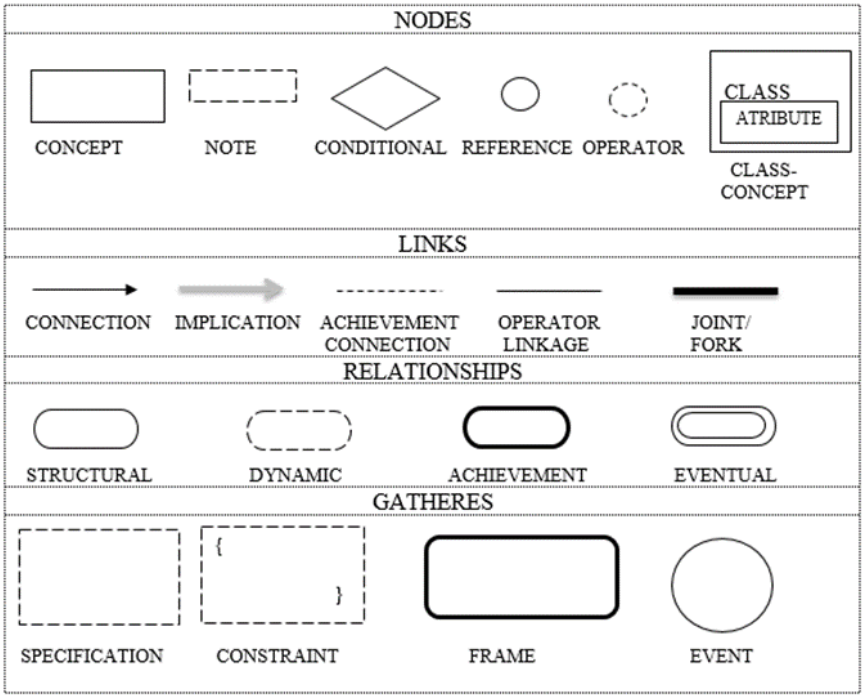
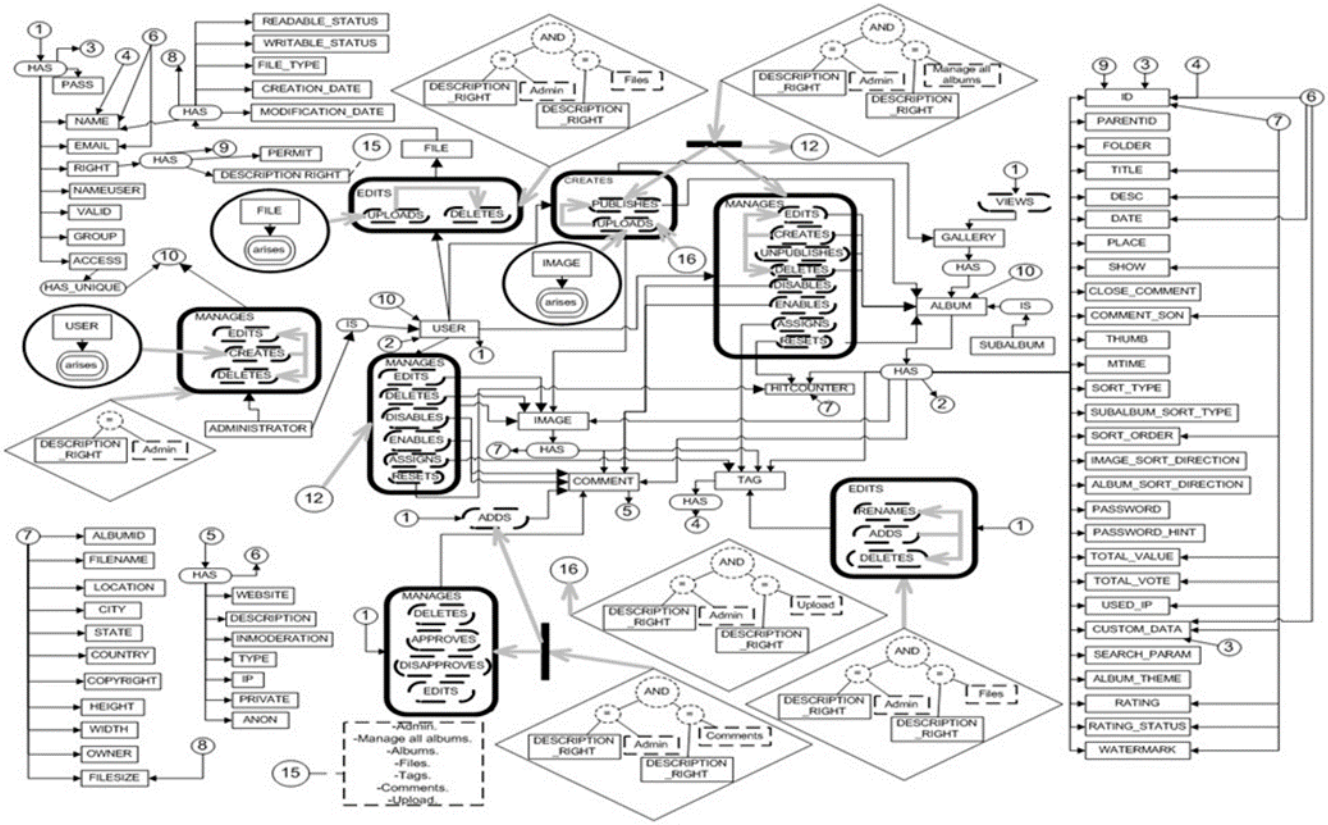


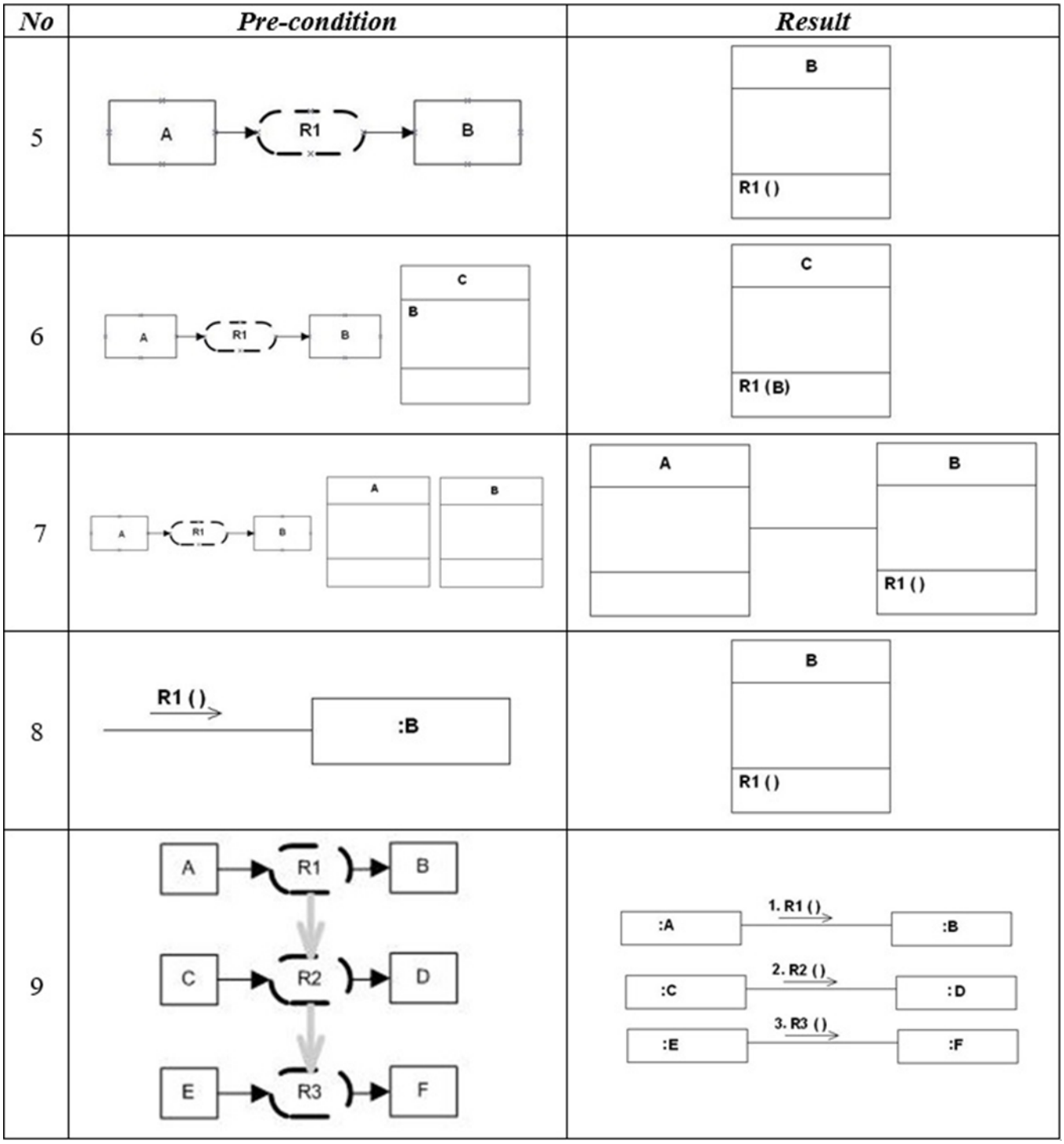
EXHIBIT 2
PRE-CONCEPTUAL SCHEMA INCLUDED IN THE UNC-METHOD GUIDE
(SOURCE: ZAPATA, 2012)



The rest of the symbols emerged from subsequent versions of the pre-conceptual schemas, starting by the notes (possible values of the concepts), achievement relationships (achievement verbs to represent goals), frames (elements for gathering together some elements or even entire pre-conceptual schemas), specifications (detailing the contents of dynamic relationships), constraints (restrictions to elements like concepts and dynamic relationships), and finally events and eventual relationships (expressing events). Zapata (2012) incorporates pre-conceptual schemas into the so-called UNC-Method, the software development method used by the National University of Colombia for training the future software engineers of Colombia. The pre-conceptual schema included in the UNC-Method is depicted in Exhibit 2.

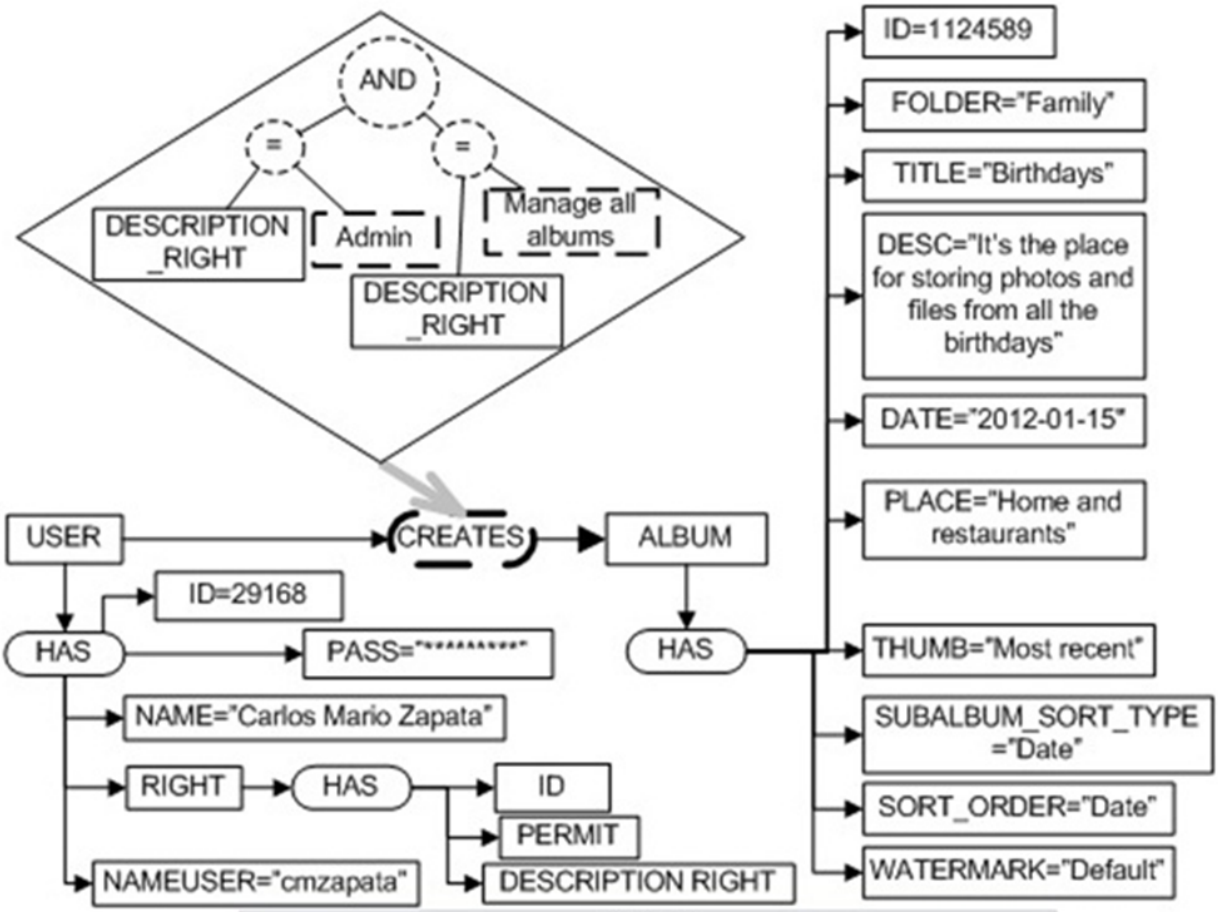
Zapata *et al.* (2006) propose some rules for automatically generating class, communication, and state machine diagrams from pre-conceptual schemas as exemplified in Exhibit 3.

EXHIBIT 3
RULES FOR AUTOMATICALLY OBTAINING CLASS AND COMMUNICATION DIAGRAMS (SOURCE: ZAPATA *ET AL.*, 2006)



Zapata *et al.* (2011) improve pre-conceptual schemas by adding exemplification to the concepts, in order to show the data needed by a dynamic relationship in order to generate the results of the same relationship. Such improvement is important for animating the specifications defined by using pre-conceptual schemas and therefore improving the stakeholder-analyst communication. Executable pre-conceptual schemas are exemplified in Exhibit 4.

EXHIBIT 4
EXAMPLE OF EXECUTABLE PRE-CONCEPTUAL SCHEMAS (SOURCE: ZAPATA, 2012)



RIGHT

ID USER	ID	DESCRIPTION RIGHT
29168	1139	Admin
29168	1140	Manage all albums

SURVEY RELATED TO PRE-CONCEPTUAL SCHEMAS FOR TEACHING SOFTWARE ENGINEERING

Several generations of software engineers have been taught by using the so-called UNC-Method. The way they use pre-conceptual schemas during the course "requirements engineering" has been considered one of the main advantages of the UNC-Method. However, this fact has never been explicitly ratified by software engineers. For this reason, we design a survey with several questions related to the experience of using pre-conceptual schemas as a way to learn software development. The size of the sample is calculated by defining a population of 800 software engineers taught by using the UNC-Method, 95% of confidence level, and 20 as confidence interval. As a result, we have a sample of 23 software engineers. They were asked about the kind of work they currently have and the results are summarized in Exhibit 5. We selected software engineers coming from several generations. The distribution of the individuals in terms of the year they started to use pre-conceptual schemas is depicted in Exhibit 6. In addition to the version they know, 4 out of 8 individuals use the 2011 version of the pre-conceptual schemas. Consequently, 19 out of 23 use the same version of the pre-conceptual schemas they know.

EXHIBIT 5
KIND OF WORK THE SAMPLE INDIVIDUALS HAVE (SOURCE: THE AUTHORS)

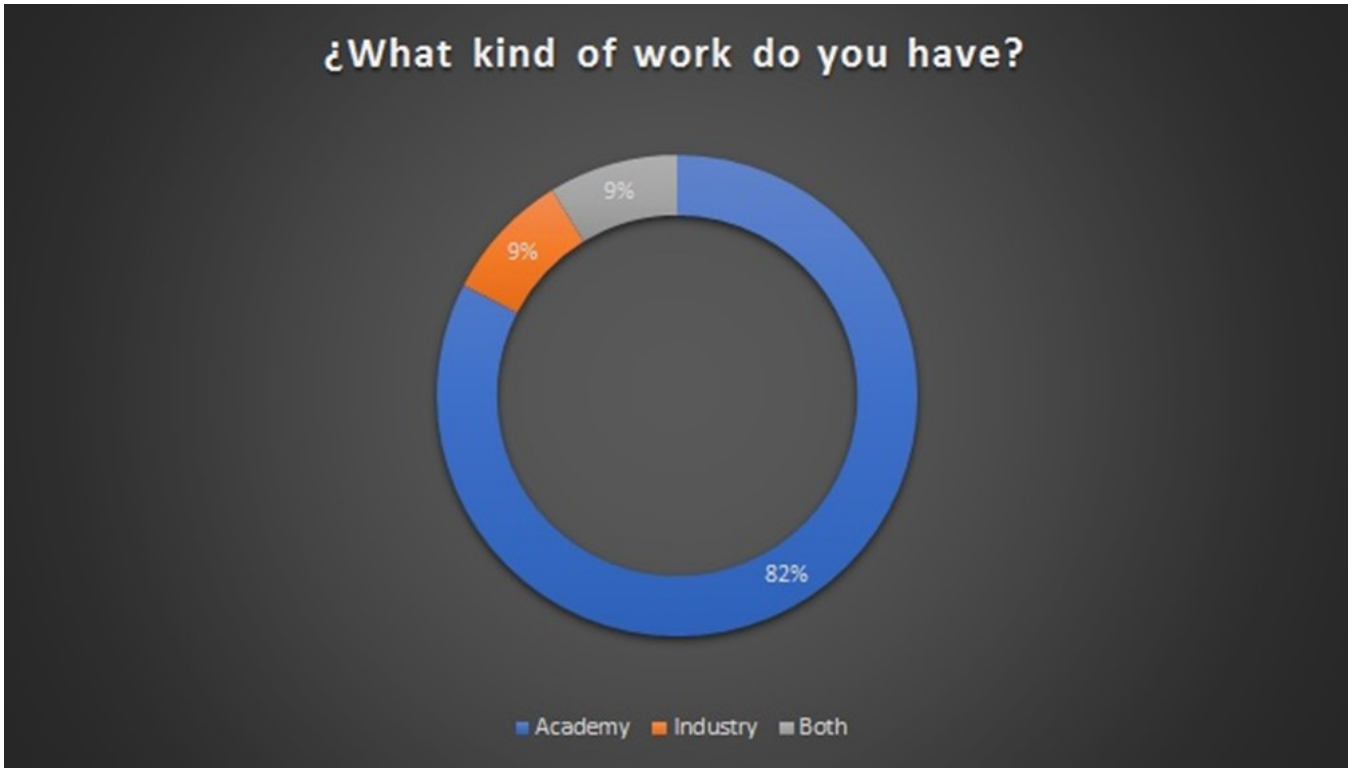
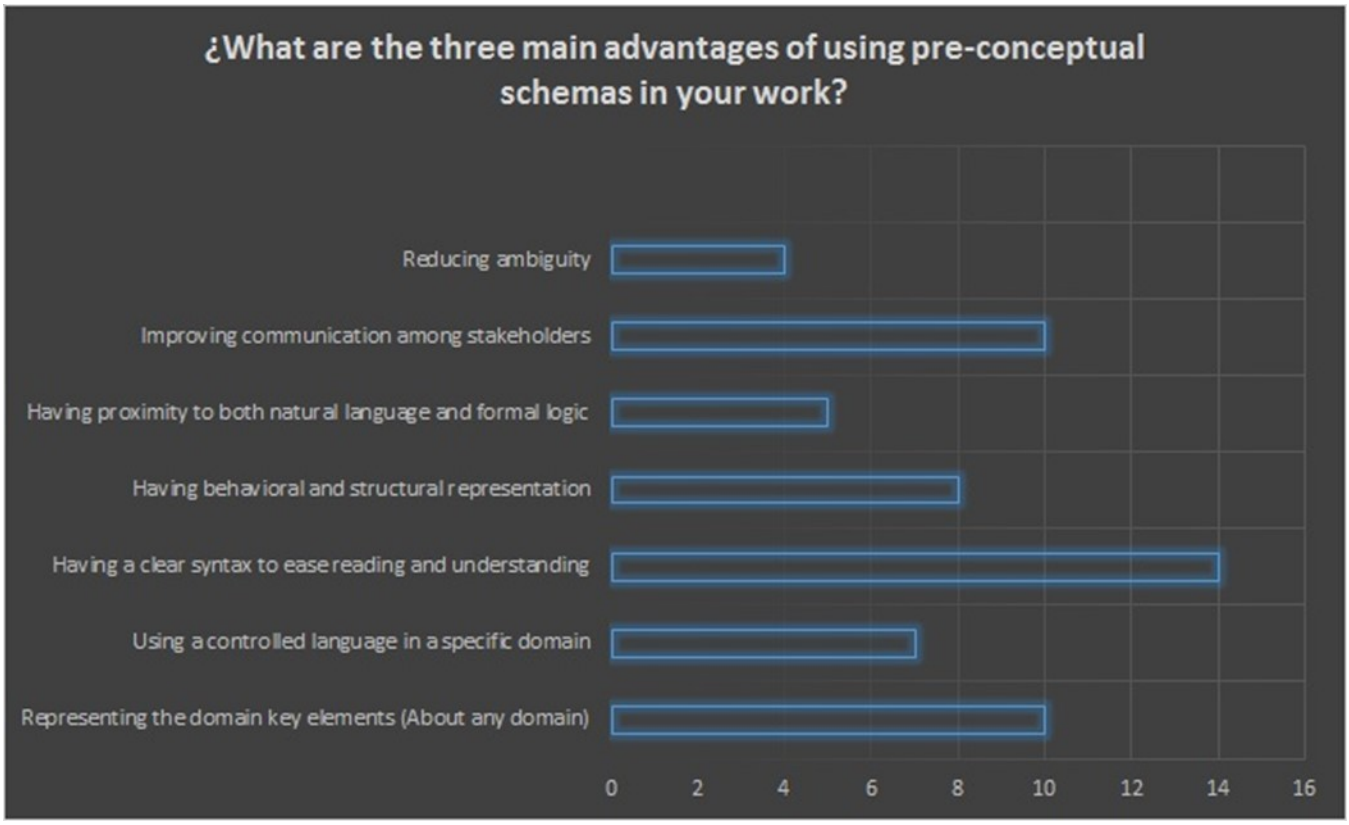


EXHIBIT 6
YEAR THE SAMPLE INDIVIDUALS STARTED TO USE PS (SOURCE: THE AUTHORS)

EXHIBIT 7
MAIN ADVANTAGES OF USING PRE-CONCEPTUAL SCHEMAS (SOURCE: THE AUTHORS)



Then, software engineers were asked about the three main advantages of using pre-conceptual schemas. Results are summarized in Exhibit 7.

In addition, software engineers were asked whether they recommend pre-conceptual schema usage to their colleagues or not, and 100% of them asked positively. The main reasons they argue for recommending pre-conceptual schemas are summarized in Exhibit 8.

Finally, sample individuals were asked about elements the pre-conceptual schemas need to be complete. Results are summarized in Exhibit 9.

RESULTS

Most of the sample individuals belong to the Academy, since most of them are currently working as software engineering professors in several universities. Feedback from software engineers working in the Industry is difficult to obtain. However, such individuals are useful for our purpose, since they are currently using pre-conceptual schemas in their classrooms. We gain feedback from individuals starting to use most of the versions released from 2007 to 2016. From 2011, executable pre-conceptual schemas have been one of the most important improvements for using pre-conceptual schemas.

Surveyed software engineers recognize the clear syntax and readability as the main advantage of using pre-conceptual schemas. In fact, this is the postulate for creating such schemas, since we need tools for communicating with stakeholders in a language closer to the natural language they speak. The next two advantages are also related to communication with stakeholders and representation of as many domain elements as we can. The fourth advantage is the completeness of the representation, since such schemas are intended to include the main structures defined by UML (structure, behavior, and interaction) but mixed in the same schema. The fifth advantage refers to the controlled language included by pre-conceptual schemas, which makes them readable. Proximity to both natural language and formal logic is the sixth advantage recognized by the sample individuals. These advantages are complemented by the seventh one, since the controlled language and the formal logic are useful for reducing ambiguity.

The reasons to recommend the pre-conceptual schemas to other colleagues seem to be equally important for the sample individuals. Such reasons can be summarized as representation capabilities of the pre-conceptual schemas. Again, modeling the structure, behavior, and interaction, and the possibility to represent knowledge of any domain are advocated as facilitators to the requirements elicitation process.

EXHIBIT 8
REASONS TO RECOMMEND PRE-CONCEPTUAL SCHEMAS (SOURCE: THE AUTHORS)

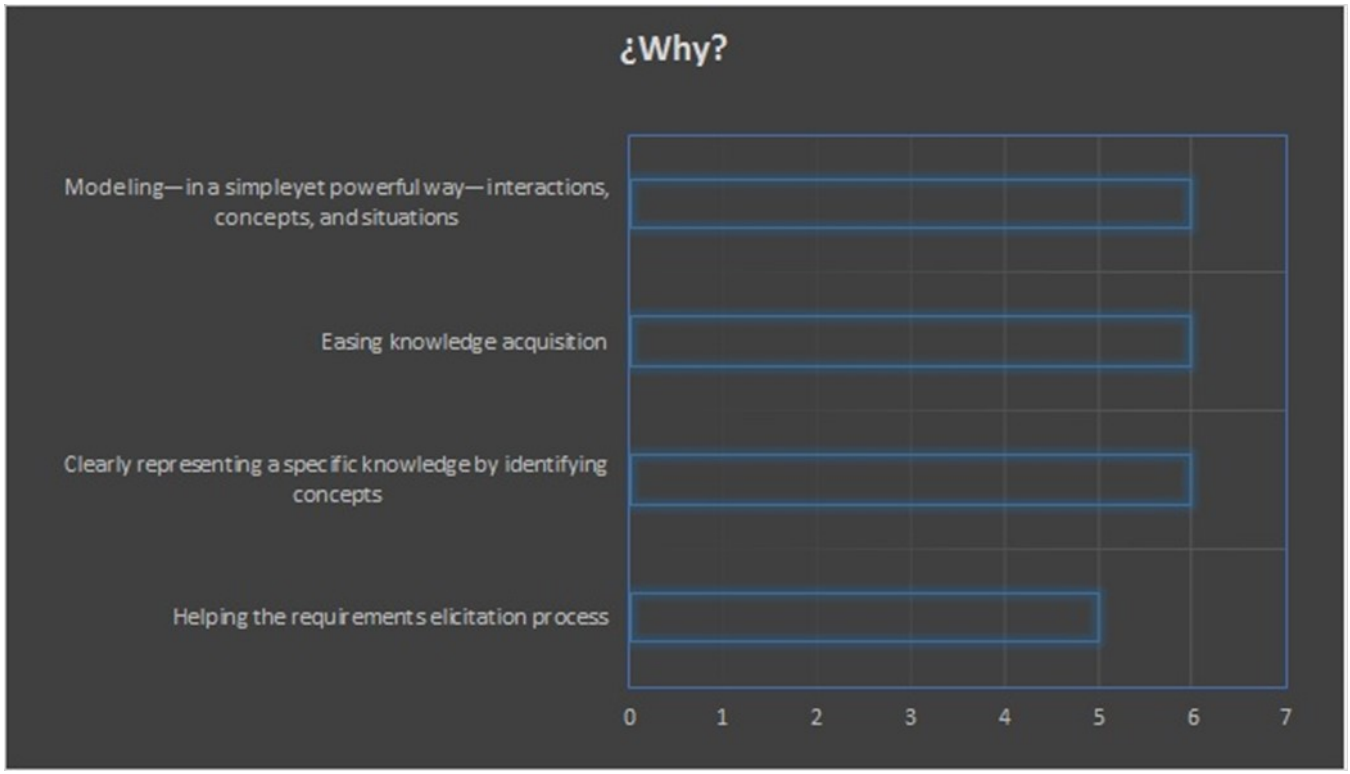
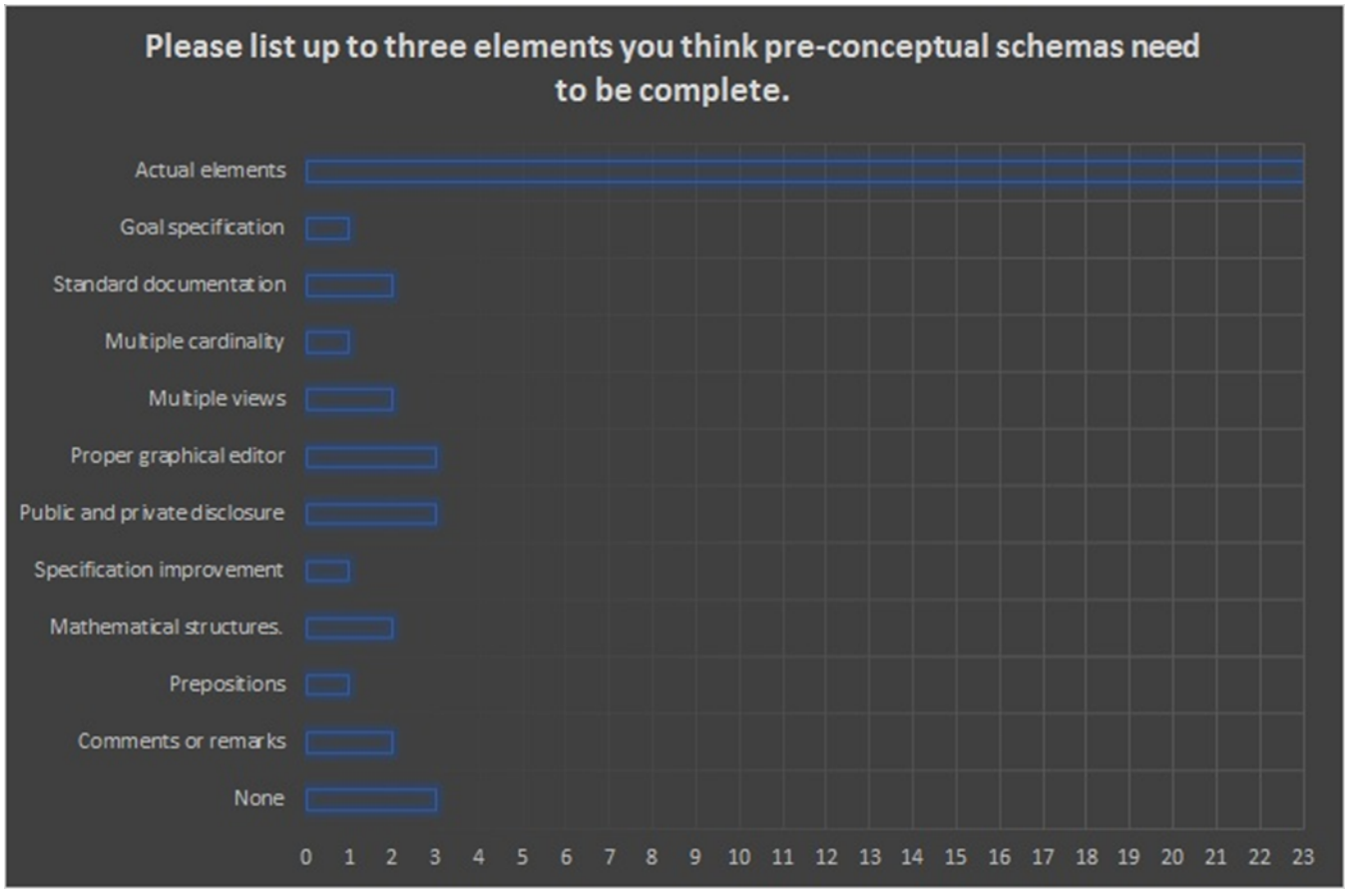


EXHIBIT 9
ELEMENTS NEEDED BY THE PRE-CONCEPTUAL SCHEMAS IN ORDER TO BE COMPLETE (SOURCE: THE AUTHORS)



Related to the elements pre-conceptual schemas need to be complete, the most common answer (actual elements of the schemas) can be explained by the heterogeneity of the years sample individuals started to use pre-conceptual schemas. Most of the mentioned elements (frames, transformation rules, events, and extensibility options) are currently included in the pre-conceptual schema specification, but most of the sample individuals ignore such elements since they use the same version from the beginning. The other elements are good suggestions for future work. For example, software engineers need a graphical editor for drawing such schemas, a standard specification of the pre-conceptual schemas, and also more disclosure is needed in Industry and other Universities, even though several professors are teaching such schemas in their universities. Some elements are currently in process, for example goal specification, mathematical structures, and specification improvement, and they will be exhibited in future versions of the pre-conceptual schemas. Some of the suggested elements are opposite to the nature of the pre-conceptual schemas, for example, multiple cardinality, multiple views, and comments. We need more insights in order to include such elements in future versions of the pre-conceptual schemas.

CONCLUSIONS AND FUTURE WORK

Pre-conceptual schemas are knowledge representations close to the natural language used by stakeholders. Such schemas are also the basis for requirements elicitation, one of the first phases of the software development process—followed by using the UNC-Method in the National University of Colombia.

As a way to support some subjective opinions about the pre-conceptual schemas, in this paper we made a survey among 23 software engineers with knowledge about the usage of pre-conceptual schemas. The main results are related to the advantages for using pre-conceptual schemas for modeling software applications: proximity to both natural language and formal logic, usage of controlled language, ambiguity reduction, and communication facility with stakeholders.

Future work is related to build a graphical editor for easing pre-conceptual schema drawing, write standard documentation about such schemas, and more publicity campaigns in order to spread the usage of the schemas. We also need to study the inclusion of some other elements, like multiple cardinality and multiple views, opposite to the nature of pre-conceptual schemas.

REFERENCES

Booch, G., Maksimchuk, R., Engle, M., Young, B., Conallen, J., & Houston, K. (2007). *Object-Oriented Analysis and Design with Applications, 3rd Edition.* Boston MA: Addison-Wesley Professional.

Fowler, M. (2004). *UML Distilled: A brief guide to the standard object modeling language.* Boston MA: Addison Wesley.

Haumer, P., Jarke, M., Pohl, K., & Weidenhaupt, K. (2000). Improving reviews of conceptual models by extended traceability to captured system usage. *Interacting with computers,* 13, 77-95.

Zapata C. M., Gelbukh, A., & Arango, F. (2006). Pre-conceptual schemas: a conceptual-graph-like knowledge representation for requirements elicitation. *Lecture notes in computer sciences,* 4293, 17-27.

Zapata C. M. (2012). *UNC-Method revisited: elements of the new approach.* Saarbrücken: Lambert Academic Publishing.