Knowledge Based Concept Analysis Method using Concept Maps and UML: Security Notion Case

Miquel Colobran, and Josep M. Basart

***Abstract*—**One of the most ancient humankind concerns is knowledge formalization i.e. what a concept is. Concept Analysis, a branch of analytical philosophy, relies on the purpose of decompose the elements, relations and meanings of a concept. This paper aims at presenting a method to make a concept analysis obtaining a knowledge representation suitable to be processed by a computer system using either object-oriented or ontology technologies. Security notion is, usually, known as a set of different concepts related to "some kind of protection". Our method concludes that a more general framework for the concept, despite it is dynamic, is possible and any particular definition (instantiation) depends on the elements used by its construction instead of the concept itself.

***Keywords*—**Concept analysis, Knowledge representation, Security, UML.

I. Introduction

FORMALIZING knowledge is an ancient problem. In the fourth century BC Aristotle included logic in his philosophical system and then the concept was understood as the intellectual representation of an object. The Aristotelian logic remained almost unchanged until the sixteenth century with the work of Leibniz [1] who began to include symbolic notation in logic. In the early nineteenth century, through the work of authors such as Boole [2], logic is related to mathematics through a mathematical system for modeling logical operations and accordingly a concept is a set of logic notions together with a set of rules. The acquisition of concepts has been a topic of study in psychology [3] and even recently, some computer science works focus on the concept notion [4].

Concept Analysis, a branch of analytical philosophy, aims at decomposing the elements, relations and meanings that compose a concept. There are several methods such as the Wilson's method [5], the Rodgers evolutionary method [6] or the Walker and Avant model [7]. Obtaining the characteristics of a concept is similar to requirement gathering or knowledge elicitation used in Computer Science. Our concept analysis is made with knowledge acquisition with constrains located into the knowledge domain and the knowledge sources. The former is reduced to a concept and the latter appears because of the difficulty to reach experts in the proposed domain.

If Concept Analysis techniques [5],[6],[7] are designed to have a clear and accurate definition of the concept under study. Usually, a concept is taken from a set of sources and, by means of several steps, how it operates and which relations it has with other concepts is revealed. The goal is to obtain a better understanding of the concept. Those techniques are particularly valuable when a concept has more than one meaning. The methods can vary according to the number of steps or the sources used. Some of them are language based and others literature based. The outcome is a language based description. Those methods are stepwise, and any enlargement of the concept or source later made implies redoing the whole analysis. Besides the methods are not suitable to be used in any computational system because they are not formal. There is neither model nor relationship between the elements and there is no detail on the constituents of the elements.

The proposed approach is a 7 step incremental and literature based method aiming at obtaining an outcome suitable to be used in object oriented engineering or ontology technologies. The objective is achieved by means of knowledge elicitation and visual modeling techniques. Knowledge elicitation is used to extract the relevant parts of text related to the concept under study. Concept maps help us to graphically represent the requirements and the Unified Model Language (UML) allows us to show graphically the elements and relations underlying the concept. The outcome reveals the attributes (the value) and behavior (as with what other concepts is related) of these concepts. The resulting graphic (a class diagram) shows these elements. Using UML as a knowledge representation language, further implementation is facilitated. Furthermore, the fact of being incremental allows the enlargement of the model adding new sources, with no need of redoing the former analysis.

This paper is organized as follows: Firstly, an overview of the techniques used to develop the method are introduced briefly after the Introduction. Secondly, the Knowledge Based Concept Analysis (KBCA) method is presented. Thirdly, a case study with the security notion.

II. Techniques Overview

Several techniques, briefly described, are used in order to obtain the proposed method (Fig: 1).

M. Colobran collaborates with the Department of Information and Communication Engineering, Universitat Autonoma de Barcelona, 08193 Bellaterra, Spain (e-mail: miquel.colobran@uab.cat).

Josep M. Basart is with the Department of Information and Communication Engineering, Universitat Autonoma de Barcelona, 08193 Bellaterra, Spain (e-mail: josepmaria.basart@uab.cat).

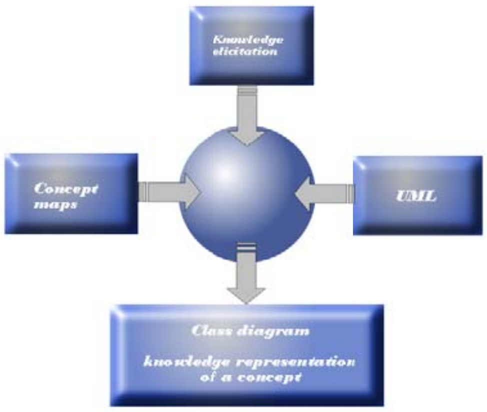


Fig. 1 KBCA Technologies involved

A. Concept Analysis

Concepts are multifaceted, abstract representations of reality [8]. Because of that, the concept analysis deals with its vagueness, ambiguity and context in order to clarify its meaning. The formal theories of concepts tries to systematize the way a concept is described such as Unified Concept Theory [4] or Formal Concept Analysis [9].

Concepts, under the view of knowledge, are a cognitive unit of meaning sometimes defined as a "unit of knowledge" (concept describes an abstract idea). The mental concepts describe a class or category. The grouping process is done by relating the aspects and qualities common to many objects. The set of all concepts gives us a representation of the world. Thus, a search for methods to handling concepts, the elements of concepts and the relations among concepts, is inevitable. Most of the used methods, such as Conceptual Maps [10], Formal Concept Analysis [11], Object Oriented techniques [12],[13],[14] or Knowledge engineering techniques [15] begin at their first stages with some sort of analysis. This analysis, in the knowledge field, is called knowledge acquisition.

B. Knowledge Acquisition

Knowledge acquisition is the process of achieving knowledge from a human expert or a group of experts [16]. The goal in knowledge engineering is the creation of knowledge-based systems (KBS).

Under the view of computer science, knowledge acquisition is a step in knowledge engineering. Broadly speaking, the knowledge life cycle include acquisition, design and implementation. Thus, software engineering and software knowledge have common points [15].

Despite there is a range of knowledge acquisition techniques [17], they deal with particular problems. Getting knowledge is made with informal methods such as interviews, questionnaires or unstructured sources, usually but not necessarily, in text form. Communication appears as a big trouble because experts and knowledge engineers have poor understanding of each other's knowledge area. Usually, experts are not able to express exactly the knowledge and therefore it is difficult to get an overview of the problem to be solved. Besides, this process has a big quantity of informal knowledge which needs to be classified, organized and formalized somehow. In short, the expert has no knowledge on knowledge engineering and the knowledge engineer has poor knowledge on expert knowledge areas.

Knowledge engineer faces other problems, such as Knowledge validation and Knowledge representation. The former is the way to verify if the knowledge is right understood and the latter the way the knowledge is expressed in order to be used to implement the system. To lighten the problem, as in software engineering, Knowledge elicitation requires tools in order to manage requirements. Usually the tools could be a simple spreadsheet, a database or requirement management system.

Mainly, obtaining requirements or knowledge is based on natural language and this presents unique difficulties [18], [19]. Many of the activities involved are cognitive and require creativity as well as knowledge about information technologies and the application domain. Several tools have appeared in order to ease the problem and try eliminating ambiguities. These try to somehow make an interpretation of natural language in order to apply the heuristics [20]. The purpose of some of these tools is to intend to minimize as much as possible the analyst's personal influence. These are still leaving the final decision of construction schemes in the analyst hands.

Thus, good knowledge gathering relies on the ability of analysts to interpret the model expressed by the user and then be able to express it in a formalized form. In software engineering, Abbot [21] first proposed a technique to gather requirements from texts. One of the big advantages to work with natural language is that it forces the developer to work on the vocabulary and space of the problem. Knowledge gathering stage, thus, is not rigorous because the natural language is ambiguous.

The final result of the knowledge elicitation step is a requirements knowledge representation. It needs to be simple to understand and formal enough to be used as the input of the knowledge implementation stage. That could be achieved by means of knowledge representation and modeling languages.

C. Mind Maps and Concept Maps

A task of concept classification somehow could be achieved using automated tools. Visual classification tools make it easy to classify objects and organize concepts. Currently, different categories of visual tools can be found [22],[23] such as Mind maps, Conceptual diagrams, Visual metaphors, Tree Maps, Flow Maps or Compare and Contrast Maps.

The most well known techniques are concept maps and mind maps. Concept maps are a way to visualize the mental "map" of concepts and their relationships, as well as the structure and hierarchy of these relationships. One important aspect of concept maps is their ability to show large amounts of information in a compact format. In this context, a concept is defined as "a perceived regularity in events or objects, or records of events or objects, designated by a label" [10].

Mind Mapping is a popular related technique devised by Tony Buzan. He describes mind maps as a net starting with a central word or concept and "around the central word you draw the 5 to 10 main ideas that relate to that word. You then take each of those child words and again draw the 5 to 10 main ideas that relate to each of those words" [24].

D. UML as a Knowledge Representation Language

Visual modeling started in Object Oriented software development methodologies and different methodologies for modeling have existed. But with no doubt, the Unified Modeling Language (UML) closed the discussion.

UML is the modeling language for software systems most well known and used today and is a de facto industry standard approved by the OMG (Object Management Group). UML is a set of specifications for object-oriented notation, which are composed of different diagrams that represent different stages of a software project development. UML combines techniques from data modeling, object modeling and component modeling. It can be used with all processes, along the software development life cycle. UML has synthesized the notations of the Booch method [25], the Object-modeling technique (OMT) [26] and Object-oriented software engineering (OOSE) [27] by fusing them into a single, common and widely usable modeling language.

UML is a graphical language for visualizing, specifying, constructing and documenting a system. The language focuses on the representation of a system and tells us how to create and read the models. However, nothing is said about how to create them. The latter is the goal of development methodologies. Some pros of UML could be found in [28]. The UML model consists of three classes of construction blocks, elements, relationships and diagrams. Elements are abstractions of real or fictitious things such as objects or actions. Relationships are the way how elements relate to each other. Diagrams reflect collections of elements along with their relationships.

The class diagram exhibits a set of classes, interfaces and relationships. This is the most common diagram in describing the design of object-oriented systems. In order to properly represent a system, UML offers a wide variety of diagrams to visualize the system from several perspectives and UML 2.0 includes 13 types of diagrams. As the aim of UML is to model any type of systems, not just software, it is also used as a knowledge representation language and the construction of ontologies [29], [30], [31].

III. KBCA

Our proposal uses together knowledge elicitation, concept maps and UML in order to produce a graphical representation of a concept. Knowledge elicitation, with constrains, is used for requirements gathering; concept maps are used to to produce a graphical representation of the requirements and UML is used to draw the final outcome. The method is named as Knowledge Based Concept Analysis (KBCA) of a concept.

A. Concept Analysis and Knowledge Acquisition Restrictions

In concept analysis the work is focused on a previously agreed concept. KBS work is focused on the domain defined at the beginning of the life cycle. Concept analysis ends when the concept is fully described and Knowledge engineering ends when the computational system is constructed. Knowledge engineering life cycle includes an analysis phase, but also has the design and implementation stages. In order to move closer concept analysis and knowledge engineering, the following points need to be considered.

- Knowledge engineering could fit purposes other than creating a computational system.
- Knowledge engineering life cycle involves several steps. Using the ones related to analysis and design, a knowledge model is obtained.
- Knowledge engineering domain is extremely flexible and could be as small as a concept.

In knowledge engineering, if the implementation stage is not done just a knowledge model of the domain is obtained. If the domain is a concept, the analysis and design stages will be focused just on that concept, its attributes and its relations. The result will be the knowledge model of a concept and become a type of concept analysis.

Another restriction is needed. When dealing with a concept, reaching the experts could be difficult or even not possible. Let's suppose a work focused on the Newton's concept of law of universal gravitation or the Descartes concept of mathematics. The concept description should be described on the basis of their writings or the interpretation of these concepts from other people. Thus, the best sources we can achieve are documents.

B. Method in Detail

KBCA consists of seven main steps as shown in Table I and Fig. 2. First three steps belong to the knowledge requirement gathering phase, fourth and fifth steps are the categorization (ordering phase). Sixth step makes the map of ideas/concepts/notions collected, and the last one converts the concept map into a class diagram.

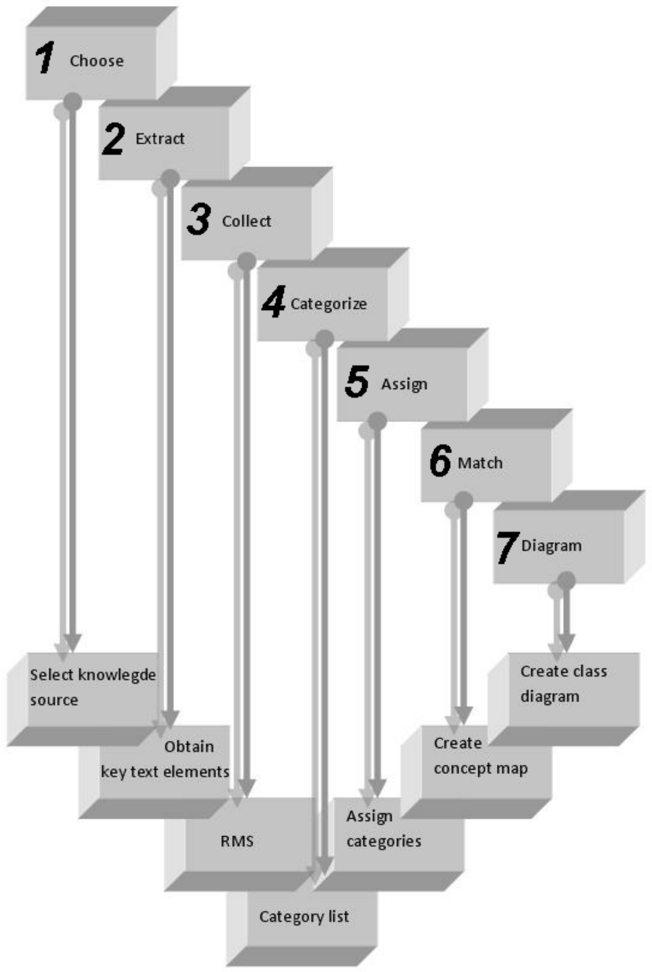


Fig. 2 KBCA Method

The relevant elements and relations are detected in the second step. This is made by emphasizing the important text pieces. That could be made in ways such as changing the text color in a word processor as well as underlining it. From now these chunks of text are called key text elements (KT_i). That could be done in parallel to populate the list or database (step 3). For clarity purposes this has been separated into two steps. This step is the most critical part of KBCA. The rest of steps rely on this one, because the final outcome heavily depends on this one being properly taken. Thus, It implies some kind of subjective component.

The third step implies collecting the information gathered into a list. That list could be made as plain text, spreadsheet, database or even a Requirement Management Software. A simple document could create about 150 key texts. Thus, a kind of mechanical tool is highly recommended. The minimal needed fields are:

- Key text number
- A way of connecting somehow the key text to the document. That could be done in many ways such as writing the key number to the document or even copying the key text into the list.
- The category

TABLE I
KBCA FLOW DIAGRAM

	Stage	Description	
Step 1	Choose	Choose knowledge source	Knowledge Elicitation
Step 2	Extract	Select key text elements	
Step 3	Collect	Insert into database and number	
Step 4	Categorize	Create list of categories	
Step 5	Assign	Assign into categories	
Step 6	Map elements	Create concept map	Concept map
Step 7	Class diagram	Construct class diagram	UML

In the fourth step, after listing, the category list is created. The possibility the category list is known from the beginning exists. Thus, the list could be created any step before. Once again, for clarity purposes, has been placed in that position.

Every requirement matches a category. This is the fifth step. The assignment is needed in order to detect extra or redundant key text. The following steps use the resulting list.

Step 6 is the most difficult part. Any key text is represented into a conceptual map. A number of situations may appear such as two similar key texts that have no relation at all or the coincidence of two key text which reveal that there is no need of rewriting. The rules to operate with key texts could be summarized as shown in table II.

TABLE II
RULES

Rule	Description	Action
$KT_i \neq KT_j \ \forall\ j$	A new element	Add to the graphic
$KT_i \supset KT_j$	Includes	No change
$KT_i \subset KT_j$	Included	No change
$KT_i = KT_j$	Same	No change
KT_i *enlarge* KT_j	The key text could add some aspect related to the previous key text	Add
KT_i	Is a relation, not an element	Add

As a result, a concept map diagram is obtained. Thus, there is a bunch of ideas and relations spread on the map.

In the seventh and last step, a conversion of the key text extraction into a more formal graphical language is made, UML. The final result is clearly understood because a standard methodology has been used.

The final diagram needs to express elements or ideas, relations and cardinality.

It is highly recommended to add some extra information at the bottom of the graphics. The reason relies on the fact that some key text elements determine values of the attributes. Thus, the list of the known values is added.

IV. Case Study of KBCA

Now, the method is applied to security concept. Barry Buzan stated that "security is a underdeveloped concept" [32] in 1983. From then, there is a plethora of work focused on what the security concept is (structure, elements, relations).

A. Applying KBCA Steps

1) Choose

The article chosen is "The concept of security" by Peter Digeser [33]. The article, unpublished but used with author permission, explores the security concept, its meanings and how it is seen by states and individuals. The work emphasizes that safety is an abstract concept and with no specification the concept is empty of meaning. Just when the elements are filled, security appears as an operational concept.

2) Extract

Text is reviewed and the text key elements are underlined as show in Fig. 3.

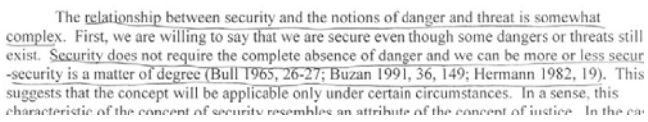

The relationship between security and the notions of danger and threat is somewhat complex. First, we are willing to say that we are secure even though some dangers or threats still exist. Security does not require the complete absence of danger and we can be more or less secur -security is a matter of degree (Bull 1965, 26-27; Buzan 1991, 36, 149; Hermann 1982, 19). This suggests that the concept will be applicable only under certain circumstances. In a sense, this characteristic of the concept of security resembles an attribute of the concept of justice. In the ca

Fig. 3 Key text elements underlined in the source

At this step just the text that looks relevant to the concept is chosen. An initial and provisional list of categories is feasible at this point.

3) Collect

A simple database is populated with the key text elements. Fig. 4 shows the database structure and Fig. 5 some key text elements.

During this step, probably, some redundant key text elements could be discovered.

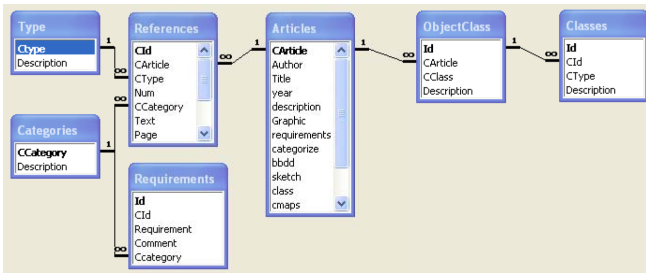
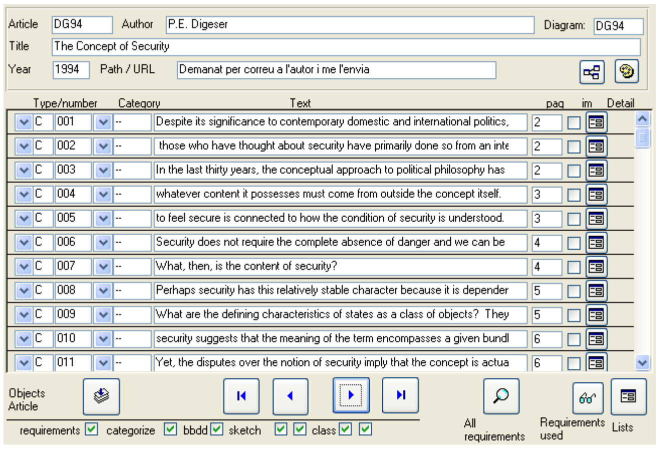


Fig. 4 Database structure


Article: DG94 Author: P.E. Digeser Diagram: DG94
Title: The Concept of Security
Year: 1994 Path / URL: Demanat per correu a l'autor i me l'envia

Type/number		Category	Text	pag	im	Detail
C	001	--	Despite its significance to contemporary domestic and international politics,	2	☐	
C	002	--	those who have thought about security have primarily done so from an inte	2	☐	
C	003	--	In the last thirty years, the conceptual approach to political philosophy has	2	☐	
C	004	--	whatever content it possesses must come from outside the concept itself.	3	☐	
C	005	--	to feel secure is connected to how the condition of security is understood.	3	☐	
C	006	--	Security does not require the complete absence of danger and we can be	4	☐	
C	007	--	What, then, is the content of security?	4	☐	
C	008	--	Perhaps security has this relatively stable character because it is depender	5	☐	
C	009	--	What are the defining characteristics of states as a class of objects? They	5	☐	
C	010	--	security suggests that the meaning of the term encompasses a given bundl	6	☐	
C	011	--	Yet, the disputes over the notion of security imply that the concept is actua	6	☐	

Objects Article
requirements ✓ categorize ✓ bbdd ✓ sketch ✓ ✓ class ✓ ✓

Fig. 5 Key text elements

4) Categorize

Once all the requirements are collected, the list of categories needs to be made. Typically there are just a few categories. If previously a provisional list has been made, it is used to make the final one. In this case, the categories listed in table III are discovered.

TABLE III
Security Categories

Categories
National Security
Security – concept
Security – attributes
Security – risk
Security – sort

5) Assign

Assigning a category to a requirement helps later on the graphical stage. We have obtained 81 key text elements. The ones which are useful to our purpose are discovered and categorized (table IV).

TABLE IV
Key Text Categorized

Categories	Quantity
National Security	4
Security – concept	40
Security – attributes	4
Security – risk	2
Security – sort	6

6) Concept Map

This is the most "traditional" step. Once the requirements are collected and organized, we have all the ingredients to create the concept map. This step involves reviewing all the requirements, one by one, in order to raise all the relationships between them and related concepts.

The relations are spread all over the text, and so are into the requirement list. The concept map may contain redundancies, i.e. the same concept appears in different requirements with, apparently, no relation or two concepts are linked together in different places.

This step is the first one that reduces the amount of information gathered. Using the mentioned rules in table II or even making new ones should be useful to create the concept map.

The outcome is a set of ideas spread onto the canvas. A lot of redundancy is eliminated as shown in Fig. 6.

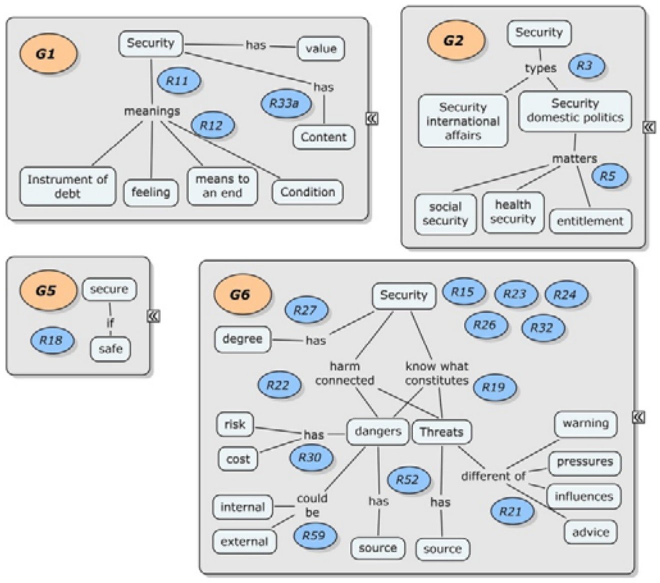


Fig. 6 Some concept map elements of the security concept

7) Class Diagram

This step helps reducing the amount of ideas in the previous stage. The outcome is a class diagram that represents the elements and relations involved (Fig. 7). The class diagram and the elements, using UML terminology, are classes and relations between classes and subclasses.

In order to create the class model, the following actions help.

- Fit each element (concept) in a class box.
- Add the attributes and behavior into the class.
- Create the relationship between elements. Add cardinality.

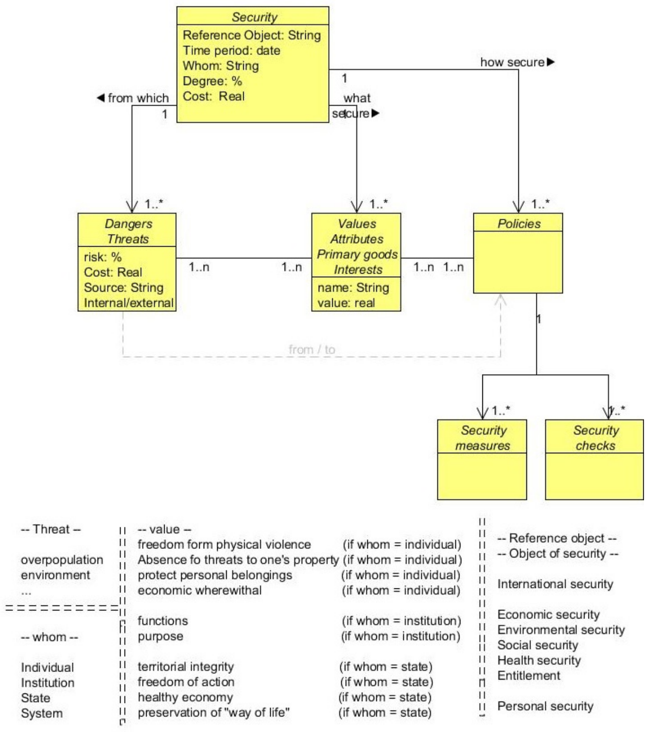


Fig. 7 Class diagram of the security concept

B. Discussion

Several points emerge from this work.

1) Meaning

The knowledge model obtained is a description of the elements, its components and relations among them. Like UML, the model has no meaning by itself. Thus, the case study of security expresses a range of possible definitions of security that are unveiled when the model is instantiated. At this point, when de components have a value, a security definition (class instantiated) appears. That security definition, using the object oriented paradigm, is unique.

2) Incremental Growth

The nature of the method permits an incremental growth of the knowledge model. An iterative process on other sources leads to a bigger and more detailed knowledge model without losing the knowledge acquired from the other sources. Even, new sources produce smaller or no changes because of the model become more complete at every cycle.

3) Uniqueness

As shown, the knowledge security model is meaningless. What if there are two security instances A and B?.

If A = B then all the elements, relations and components are the same, and we can conclude that the security definition is the same.

If A and B are two security objects with B having, for example, a different set of policies or threats, we can conclude that in this scenario, A ≠ B.

Thus, there is no unique security definition. There are just security concept constructions and as many securities as different security objects we are able to create. This is the reason why the "definition" of the resulting security is different. Therefore, persons, groups or states perceive different notions of security because the defining elements vary remarkably.

Besides, if we create a different construction of security (from other source for example), all the resulting objects will be different security objects (despite being neither semantically nor in practice incorrect).

4) Security Definitions and Computer Security

From Barry Buzan work [32], a wide range of security definitions are identified. For example, the human security from UNPD [34] or the expanded notion of security stated by Emma Rothschild [35] who argued that security notion is extended in "four main forms". Open questions emerge such as is if all of those securities could be considered a kind of a bigger security model, actually a knowledge security model and how computer security and the existing securities could be peacefully integrated in such model. Because of the fact that computers are social tools, Computer Security needs an inter-disciplinary work in order to become another kind of security.

V. Conclusions Remarks

A methodology for exploring the underlying elements in a concept and the relationships among them is proposed. The outcome is an abstract concept, which requires specific elements to produce "the definition". This definition is extremely flexible and can be adapted to almost any framework of any field.

The knowledge based concept analysis (KBCA) proposed method is based on knowledge engineering, concept maps and UML. It's intended to extract knowledge from any informal source in order to obtain concept class diagram. That outcome could be used in object oriented engineering or knowledge based systems such as ontologies.

Concept analysis can also be made with knowledge elicitation applying some restrictions in the domain and the steps involved. The outcome of design stage in knowledge engineering, when the domain is restricted to one concept, leads to a type of concept analysis. The proposed method is a 7 steps concept analysis and literature based in order to overcome expert elicitation problems.

In the proposed scenario, the knowledge engineering analysis and design stages are focused just on one concept its attributes and its relations. The result is the knowledge model of a concept.

Traditional concept analysis methods are stepwise. Our proposal is incremental, thus enlarge the model is easier. The UML purpose is to model any type of systems (not just software). This language should be understandable to humans and machines and could be used as a knowledge representation language.

KBCA is very systematic. Further implementation of the result, if needed, will be easier because of UML is used. The resulting diagram could be used to check by end-users or documentmakers and even could be used to integrate in bigger projects, related or not with computer software.

Despite we have reduced as much as possible the subjective component, the requirements gathering are a human task and the method still suffers from a subjective component. Thus, most probably the same text analyzed by several people may easily lead to slightly different outcome.

International Relations field has made, in the last decades, a lot of work on the concept and structure of the security notion. Their main concern are the types of securities, the existing relationship between several securities, security policies and, to a lesser extend the semantic notion of security and its consequences on individuals, entities or nationalities. There are no works available in order to link that security with information security in computer science. A generic framework could benefit both fields.

The security concept is meaningless until all the elements are instantiated and the "definition" of security relay on the values instead of the word on its own.

In the case of complex concepts, the review from just a single source of knowledge is clearly insufficient. Therefore, a further work to obtain a class diagram (formalization of a concept) from many sources (formal or informal) is needed. In order to extend the range, other kind of sources such as written documents, voice recordings, pictures and in general any multimedia documents need to be included.

More research is also needed to discover in which areas this methodology is useful and what changes or improvements would be needed to adapt to these new scenarios.

References

[1] L. C. Agrela, "La superacion por Leibniz de la logica aristotelica," Revista Internacional de Filosofia, vol. Suplemento 3,, pp. 67–74, 2010.
[2] P. Jetli, "The Completion of the Emergence of Modern Logic from Boole's The Mathematical Analysis of Logic to Frege's Begriffsschrift," in Logic and Its Applications, ser. Lecture Notes in Computer Science, M. Banerjee and A. Seth, Eds. Springer Berlin Heidelberg, 2011, vol. 6521, pp. 105–123. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-18026-2_10
[3] R. Streveler, T. Litzinger, R. Miller, and P. Steif, "Learning Conceptual Knowledge in the Engineering Sciences: Overview and Future Research Directions," Journal of Engineering Education, vol. 97, pp. 279–294, July 2008.
[4] J. Goguen, "What Is a Concept?" Lecture Notes in Computer Science: Conceptual Structures: Common Semantics for Sharing Knowledge, pp. 52–77, 2005. [Online]. Available: http://dx.doi.org/10.1007/11524564_4
[5] J. Wilson, Thinking with concepts. Cambridge University Press, 1963.
[6] B. L. Rodgers, "Concepts, Analysis, and the Development of Nursing Knowledge: The Evolutionary Cycle." Journal of Advanced Nursing, vol. 14, pp. 330–335, 1989.
[7] K. C. Walker, L.O. Avant, Strategies for theory construction in nursing, 3rd ed. Norwalk, CT: Appleton & Lange, 1995.
[8] V. L. Griffin-Heslin and al., "An analysis of the concept dignity," Accident and Emergency Nursing, vol. 13, pp. 251–257, 2005.
[9] U. Priss, "Formal Concept Analysis in Information Science," Annual Review of Information Science and Technology, vol. 40, pp. 521–543, 2006. [Online]. Available: (http://www.upriss.org.uk/papers/arist.pdf).
[10] J. D. Novak and A. J. Cañas, "The Theory Underlying Concept Maps and How to Construct Them," Technical Report IHMC CmapTools, Tech. Rep. 2006-01, 2006. [Online]. Available: (http://cmap.ihmc.us/Publications/ResearchPapers/TheoryCmaps/TheoryUnderlyingConceptMaps.htm).
[11] B. Ganter and R. Wille, Formal Concept Analysis: Mathematical Foundations, 1st ed. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 1997.

[12] R. S. Pressman, Software engineeering: a practioner's approach, 6th ed. Boston, EUA: McGraw-Hill, 2005.
[13] M. O'Docherty, Object-oriented analysis and design : understanding system development with UML 2.0. John Wiley & Sons, 2005.
[14] P. Coad and E. Yourdon, Object-Oriented Analysis. London: Prentice-Hall, 1991.
[15] R. Studer, R. Benjamins, and D. Fensel, "Knowledge engineering: principles and methods," Data and knowledge engineering, vol. 25, pp. 161–197, 1998.
[16] S. Kendal and M. Creen, An Introduction to Knowledge Engineering, 1st ed. Springer, Oct. 2006.
[17] J. Hua, "Study on Knowledge Acquisition Techniques," in Proceedings of the 2008 Second International Symposium on Intelligent Information Technology Application - Volume 01, ser. IITA '08. Washington, DC, USA: IEEE Computer Society, 2008, pp. 181–185. [Online]. Available: http://dx.doi.org/10.1109/IITA.2008.152.
[18] S. Potter, "A Survey of Knowledge Acquisition from Natural Language," AKT project report Task 1.1.2, 2001.
[19] J. Wang, Y. Wu, X. Liu, and X. Gao, "Knowledge acquisition method from domain text based on theme logic model and artificial neural network." Expert Syst. Appl., vol. 37, pp. 267–275, 2010. [Online]. Available: http://dx.doi.org/10.1016/j.eswa.2009.05.009.
[20] S. P. Overmyer, B. Lavoie, and O. Rambow, "Conceptual Modeling through Linguistic Analysis Using LIDA." in Software Engineering, 2001. ICSE 2001. Proceedings of the 23rd International Conference. IEEE Computer Society, 2001, pp. 401–410.
[21] R. J. Abbott, "Program design by informal English descriptions," Commun. ACM, vol. 26, pp. 882–894, November 1983. [Online]. Available: http://doi.acm.org/10.1145/182.358441.
[22] L. Dillard and B. Myers, "Visual Teaching Tools: Concept Maps," University of Florida, Tech. Rep., May 2008.
[23] M. J. Eppler, "A comparison between concept maps, mind maps, conceptual diagrams, and visual metaphors as complementary tools for knowledge construction and sharing," Information Visualization, vol. 5, no. 3, pp. 202–210, 2006.
[24] T. Buzan and B. Buzan, The Mind Map Book, 2nd ed. London: BBC Books, 1995.
[25] G. Booch, Object-oriented analysis and design with applications (2nd ed.). Redwood City, CA, USA: Benjamin-Cummings Publishing Co., Inc., 1994. [Online]. Available: http://portal.acm.org/citation.cfm?id=174890.
[26] J. Rumbaugh, Object-oriented modeling and design. Prentice Hall, 1991.
[27] I. Jacobson, Object Oriented Software Engineering: A Use Case Driven Approach. Addison-Wesley, 1992.
[28] E. H. Orallo, "El Lenguaje Unificado de Modelado (UML)," Manuales Formativos ACTA, num 26, October 2002.
[29] S. Cranefield and M. K. Purvis, "UML as an Ontology Modelling Language." in In Proceedings of the Workshop on Intelligent Information Integration, 16th International Joint Conference on Artificial Intelligence (IJCAI-99), 1999, pp. 46–53.
[30] A. Felfernig, G. Friedrich, D. Jannach, and M. Zanker, "Configuration Knowledge Representation Using UML/OCL," in "UML" 2002 - The Unified Modeling Language. Springer Berlin / Heidelberg, 2002, pp. 91–108. [Online]. Available: http://dx.doi.org/10.1007/3-540-45800-X_5.
[31] C. W. Chan, "Knowledge and software modeling using UML." Software and System Modeling, vol. 3, no. 4, pp. 294–302, 2004.
[32] B. Buzan, People, States and Fear. Harvester-Wheatsheaf, Brighton, 1983.
[33] P. Digeser, "The Concept of Security," 1994, presented at the Annual Meeting of the American Political Science Association 14 September 1994. Obtained from author. Unpublished.
[34] Undp, HDR 1994 - New Dimensions of Human Security. Human Development Report Office (HDRO), United Nations Development Programme (UNDP), 1994. [Online]. Available: http://EconPapers.repec.org/RePEc:hdr:report:hdr1994.
[35] E. Rothschild, "What is security? the quest for world order," Daedalus, vol. 124, no. 3, pp. 53–99, June 1995.

Miquel Colobran is a doctoral student at the Department of Information and Communication Engineering. His research is in the field of ontologies, security and Social Computing.

Josep M. Basart is a PHD professor at the Department of Information and Communication Engineering. His research is in the field of Computer Ethics, Engineering Ethics, Applied Ethics and Social Computing.