BUILDING CLASS DIAGRAMS SYSTEMATICALLY

M.J. Escalona
Department of Computer Languages and Systems. University of Seville. Spain

J.L. Cavarero
Laboratoire CNRS / I3S, Sophia – Antipolis. University of Nice, France

Keywords: Static model, class diagram, web methodology, systematic process

Abstract: The class diagram has become more important since the object-oriented paradigm has acquired more acceptance. This importance has been translated also in the new field of web engineering. However, in a lot of cases, it is not easy to get the best class diagram in a problem. For this reason, it is necessary to offer systematic processes (as cheaper and easier as possible) to give a suitable reference to the development team. This work presents two different processes developed in the University of Nice and in the University of Seville and applies them to the same problem comparing the results and getting some important conclusions.

1 INTRODUCTION

The object oriented paradigm has been the most accepted in the last years to model software information systems. The class diagram has been assumed by, not only the research community, but also by the company environment as the best model to represent the static model of information systems.

The importance of the class diagram is also very obvious in web development (Desphande et al., 2002). In the last years, most of the groups working in web methodologies have accepted the class diagram as the way to represent the conceptual data model, that is, the static model in web environment (Barry & Lang, 2001), (Koch, 2001) (Escalona & Koch, 2004). Moreover, the developer who applies a class diagram to represent his system can very often discover that it is not easy to get a good class model. (Cachero & Koch, 2002) (Insfran et al., 2002). For this reason, the research community is looking for systematic processes, ways and heuristics, which help the developer to build the class diagram. But, the reality is quite different, if we achieve a comparative study on the actual methodologies (Barry & Lang, 2001), (Koch, 2001) (Escalona & Koch, 2004), we can deduce that they do not offer suitable, easy and cheap processes to build the class diagram from the requirement catalogue.

In this work, we present two systematic processes to get the class diagram from requirements. These processes were applied to a real project develop in Seville, which is presented in the third section, and very interesting results were obtained.

2 SYSTEMATIC PROCESSES TO BUILD CLASS DIAGRAMS

If we analyse the different proposals in software engineering and also in web engineering, we can discover that all methodologies give a great importance to this model (Barry & Lang, 2001), (Koch, 2001) (Escalona & Koch, 2004). For this reason, some research groups have worked in developing systematic processes, which help in a very easy, cheap and fast way to get this model.

Actually, there are very few systematic processes At this point, we are going to present two proposals which offer algorithms to get a class diagram in a systematic and, even, automatic way.

2.1 NDT- Navigational development techniques

NDT(Escalona, 2004) is a methodological proposal to specify and analyse web information systems. NDT development process can be defined as a

J.Escalona M. and L. Cavarero J. (2005).
BUILDING CLASS DIAGRAMS SYSTEMATICALLY.
In *Proceedings of the Seventh International Conference on Enterprise Information Systems*, pages 378-381
DOI: 10.5220/0002511103780381

bottom-up process. The development process is focused on a very detailed definition of requirements guided by objectives, which covers three subphases: requirements capture, requirements definition and requirements validation. The NDT process only covers the first phase in the life cycle. It is a bottom-up process where models are independent. Also, it is important to note that the processes to get analysis models from requirements are systematic. They can even be automatic if the development team uses NDT-Tool (Escalona et al., 2003, 2004), which is a case tool that supports the NDT development process.

2.2.1 From storage requirements to the basic conceptual model

In this work, only one of the systematic processes in NDT is presented. This process allows you to get the basic conceptual model starting with the storage information requirements definition. At this point, we are going to explain this process in a global way (more information in (Escalona, 2004)).

As we said, in NDT, each requirement is dealt in a different way depending on its type. One of the most important is the group of storage information requirements. These requirements describe the information which the system deals with. To describe this kind of requirements, NDT proposes a special pattern. In table 1, an example of this pattern is presented. The process is quite simple, each storage requirement pattern expresses which information should be stored in the system as an abstract concept. For example, in table 1, we show the information that the system has to store for each monument. One of the most important fields in this pattern is the specific data. The specific data field describes which concrete information is stored and its nature. For instance, for each monument, the system stores the name. Its nature is String. The nature can be of two kinds in NDT: the basic natures or other storage requirements. Basic natures can be String, Number, etc. When a specific data has a nature of another storage requirement, it means that the structure of this specific data is described in the pattern of this storage requirement. For instance, in table 1 the specific data "Conservation Data" has a nature SR-02. It expresses that its structure is described in the pattern SR-02.

Basically, in the systematic process, each storage pattern produces a class. The specific data with basic nature are translated into an attribute in this class. The specific data with nature of another storage requirement are translated in associations.

Table 1: A storage requirement definition.

SR-01	Information of the monument	
Associated objectives	OBJ-01: To manage the information of monuments	
Description	The system must store information about monuments. Specifically:	
Specific data	Name and description	Nature
	Name: it's the name of the monument.	String
	Address: it's the address of the monument.	String
	…	…
	Typology: it's the set of the monument typologies.	String Cardinaly:0..n
	Image: it's a set of images where the monument appears.	Image Cardinality: 0..n
	Conservation data: it stores the piece conservation studies.	SR-02 Cardinality: 0..n

2.2 The Composition method

The Composition method (Cavarero & Lecat, 2000) is a systematic process that starts with attributes and methods. It proposes 4 steps to get the distribution of these attributes and methods in classes, in the most efficient way.

The process starts by identifying the methods and attributes and by designing a matrix. The methods are in columns and the attributes are put in rows. Then, it is necessary to study how each method deals with each attribute. If a method modifies one attribute, a symbol O is included. If a method consults but not modifies one attribute, we express the relation with X. In our example, attributes and methods are identified with NDT, so it is easier to design the matrix which is presented in table 2.

Table 2: A part of the matrix.

Attribute	m1	m2	....	m15	m16
Town	X	X		X	X
City	X	X		X	X
Authors					
Characterization	X	X		X	X
Code	X	X		X	X
......					
Type	O	O			

After designing the matrix, the next step is to study the methods which modify any attribute. Each of these methods is put in one class with the attributes that they modify. In the third step, if two classes have the same attributes or if the set of attributes in one class is contained in another one, the two classes are joined. After that, classes with common attributes, but not contained, are connected. For instance, if class A has the attributes A1, A2 and class B has the attributes A1, A3, then attribute A1 is deleted in class B and in class A, a new public

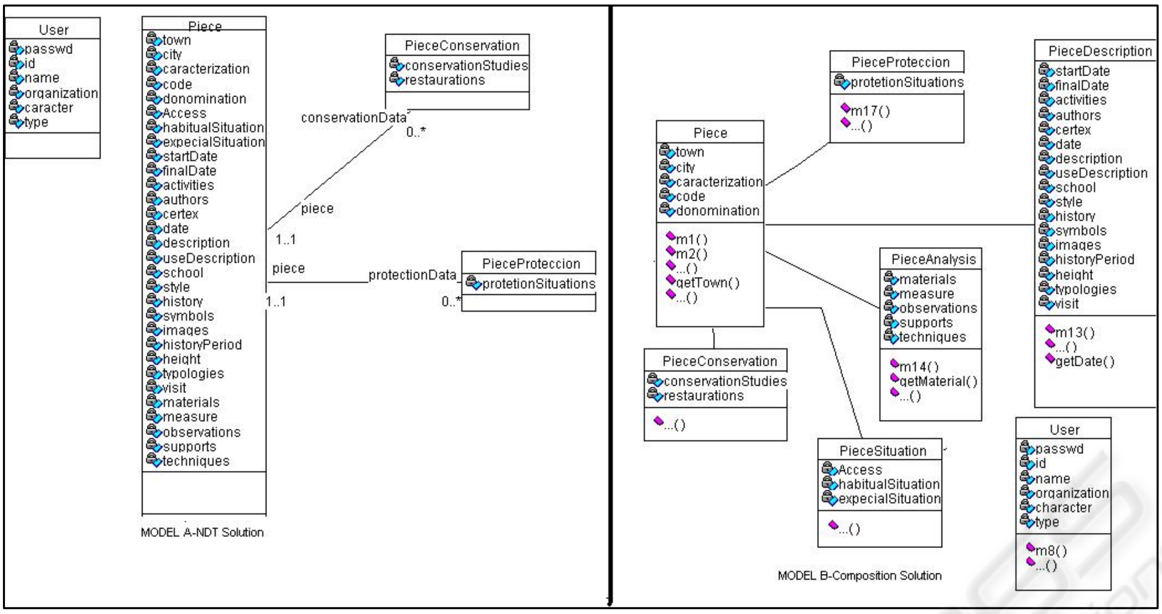

Figure 1: The result of the application of NDT and the composition method.

method, getA1(), is added (to make it possible for the method in B to read A1) and a relation between A and B is created. In the last step, we add in the diagram the methods that only consult attributes. These methods must be added by trying to get the least new relations as possible.

3 AN EXAMPLE: THE SYSTEM TO MANAGE MONUMENTS

Since the last years, a lot of people and tourists are interested in getting information about monuments and cultural heritage. In Andalusia, in the south of Spain, there is a public organization which manages the information about monuments: the Andalusian Institute of Historic Patrimony (IAPH, 2004). Some years ago, this organization started to develop the Information System of Andalusian Historical Patrimony (SIPHA, 2004). This system stores information about monuments in very different media: image, text, documents, etc. It is a very big system because in Andalusia there are a lot of monuments. The system stores about 130 attributes for each monument where we can find data like the name of each monument, the name of its authors, images of the monuments, etc. This amount of data makes it necessary to design a class diagram very efficiently to get the most suitable results.

4 BUILDING THE CLASS DIAGRAM

Starting with the example of monuments, which was completely specified by NDT, we reduced the number of attributes and methods, working only with the most significant We applied the NDT method and the Composition method on a part of this system, which contains originally 40 attributes and 16 methods. The name of these attributes and methods are quite intuitive, so we are not going to explain all of them because the most interesting issue is to show the results.

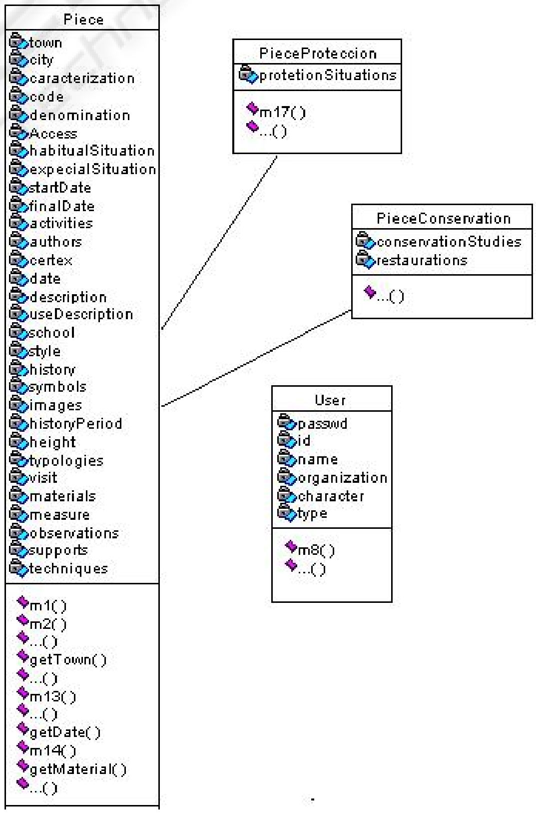

Figure 2: The final model.

In NDT the work started with the storage information requirements patterns. The problem of monuments was completely developed with NDT in (Escalona et al., 2004)). In table 1, we can observe one of them. From them the conceptual model A in figure 1 was gotten. To apply the composition method, the first task was to build the matrix. In table 2 we present a part of the matrix for this system. Applying the algorithm, we got the conceptual model B in the figure 1. After the application of the algorithms and, if we continue the process in NDT and in the Composition method, we can find that the final model is the same for both methods. The classes Piece, PieceDescription, PieceSituation and also PieceAnalysis, are joined in the model of the Composition method and the cardinalities of the other associations are the same as in NDT. In NDT, the final distribution of the methods is the same as in the Composition method. In this way, the final model in both techniques is shown in figure 2.

5 CONCLUSIONS

The necessity of offering systematic process and tools to get class diagram, in an easy and cheap way, has been detected by several research groups. This work has presented two proposals that offer a guide reference to build class diagrams.

The first one, NDT, is focused in patterns, specific data and relations. The second one is focused in methods and attributes. After applying both proposals, it is very interesting to observe that the results are quite similar, although they are focused in different aspects. Moreover, the final result in each of them is the same and also similar to the real solution found in the IAPH.

These similar results allow us to guarantee that both proposals are based on correct processes, they only offer different ways to build class diagrams.

The application of both models is not difficult. Each of them offers tools that allow us to apply them automatically. For these reasons, in very complex systems, with a lot of attributes, methods, uses cases or storage requirements, it could be interesting to apply both of them and compare the results.

REFERENCES

Barry, C. & Lang, M. (2001) A Survey of Multimedia and Web Development Techniques and Methodology Usage. IEEE Multimedia. April-June 2001, 52-56.

Booch, G., Rumbaugh, J., Jacobson, I. Unified Modeling Language User Guide. Ed. Addison-Wesley, 1999.

Cachero, C. , Koch, N. Conceptual Navigation Analysis : a Device and Plataform Independent Navigation Specification. 2nd IWOST. June, 2002.

Cavarero, J.L, Lecat, R. La conception orientée object, évidence ou fatalité. Technosup. Les filiéres technologiques des enseignements supérieurs. Ed. Ellipses. France 2000.

Deshpande, Y., Marugesan, S., Ginige,A., Hanse,S., Schawabe,D., Gaedke, M, B. White. Web Engineering. Journal of Web Engineering. V. 1 N 1. 2002. Rinton Press

Durán A., Bernárdez, B., Ruiz, A., Toro M. (1999). A Requirements Elicitation Approach Based in Templates and Patterns. Workshop de Engenharia de Reqisitos. Buenos Aires, Argentina.

Escalona, M.J. Modelos y técnicas para la especificación y análisis de la navegación en sistemas software. Ph-European Thesis. Dpto. Lenguajes y Sistemas Informáticos. 10/2004. Sevilla, Spain.

Escalona, M.J., Koch, N Requeriments Engineering for Web Applications: A Comparative Study. Journal of Web Engineering". pp.193-202. Volumen: 2, Nº·3 2004. Rinton Press. New Jersey, Febrero 2004.

Escalona M.J, Mejías M, Torres J, Reina A.M. NDT-Tool: A tool case to deal with requirements in web information systems. Proceedings of IV International Conferences on Web Engineering. ICWE 2003. LNCS 2722. 2003.

M Escalona M.J, Mejías M, Torres J. Developing systems with NDT & NDT-Tool. 13th International Conference on Information Systems Development: Methods and Tools, Theory and Practice. pp.149-159. Vilna, Lituania. Septiembre 2004.

Insfrán, E., Pastor, O., Wieringa, R. Requirements Engineering-Based Conceptual Modeling. Requirements Engineering Journal, Vol 7 (1). 2002.

IAPH, Instituto Andaluz de Patrimonio Histórico. Dirección General de Bienes Culturales. Consejería de Cultura. Junta de Andalucía. http://www.juntadeandalucia.es/cultura/iaph

Koch, N. Software Engineering for Adaptative Hypermedia Applications. Ph. Thesis, FAST Reihe Softwaretechnik Vol(12), Uni-Druck Publishing Company, Munich. Germany. 2001.

Rossi, G. (1996) An Object Oriented Method for Designing Hipermedia Applications. PHD Thesis, Dpto. de Informática, PUC-Rio, Brazil, 1996.

Rumbaugh, J. OMT: Modelation and conception object oriented. Masson, 1994.

SIPHA. El Sistema de Información del Patrimonio Histórico de Andalucía. http://www.juntadeandalucia.es/cultura/iaph/documentacion/documentacion.html