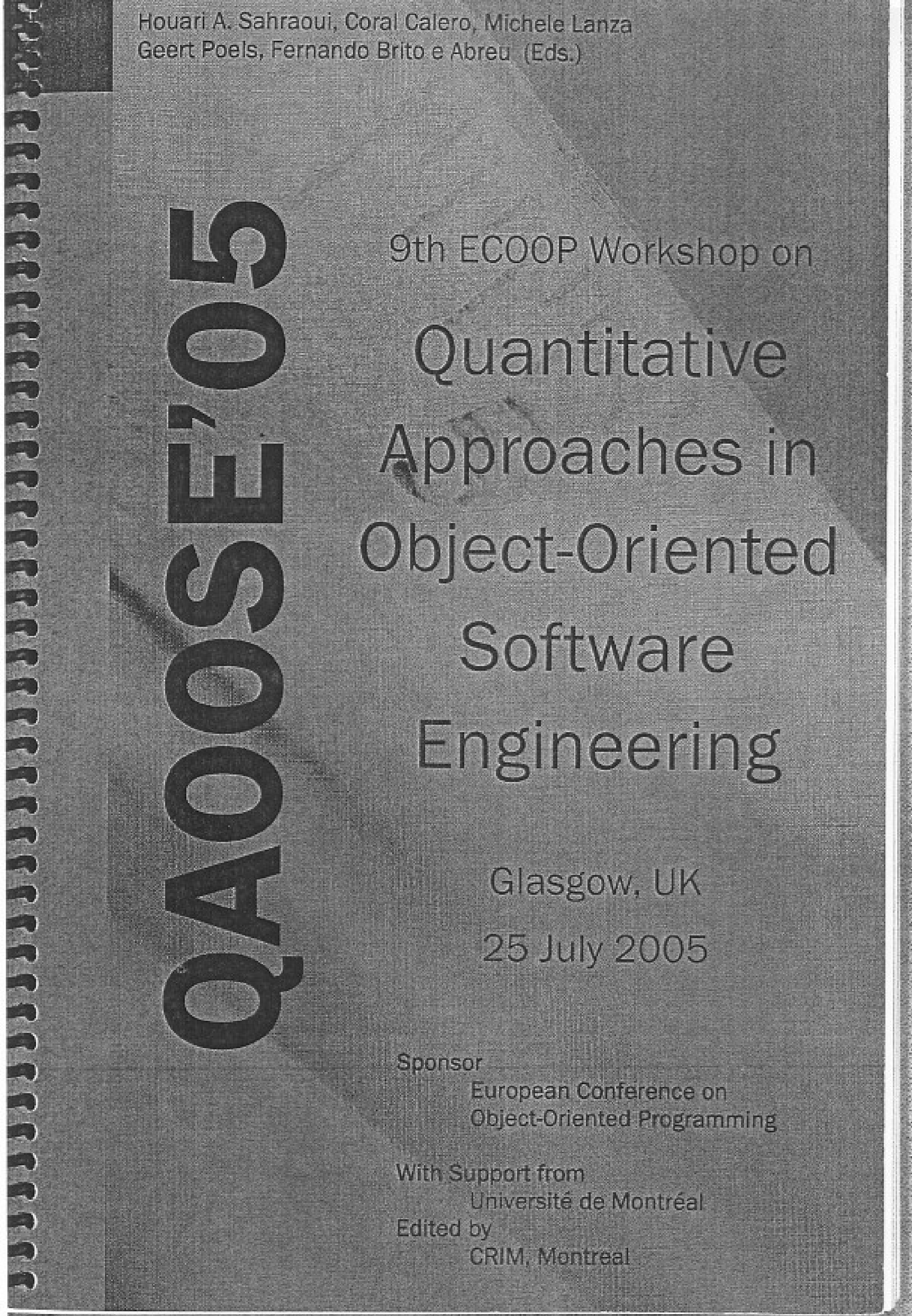

Houari A. Sahraoui, Coral Calero, Michele Lanza
Geert Poels, Fernando Brito e Abreu (Eds.)

QAOOSE'05

9th ECOOP Workshop on

Quantitative Approaches in Object-Oriented Software Engineering

Glasgow, UK

25 July 2005

Sponsor
European Conference on
Object-Oriented Programming

With Support from
Université de Montréal
Edited by
CRIM, Montreal

Edited by
CRIM, Montreal

ISBN 2-89522-065-4

Sponsors

ecoop
2005
European Conference
on Object-Oriented Programming

Université de Montréal

Escuela Superior de Informatica
Universidad de Castilla-La Mancha
http://www.inf-cr.uclm.es

USI

UNIVERSITEIT GENT

nova

CRIM

Organization

Houari A. Sahraoui
sahraouh@iro.umontreal.ca
Département d'Informatique et Recherche Opérationnelle
Université de Montréal
CP 6128 succ Centre-Ville, Montréal QC H3C 3J7, Québec, Canada

Coral Calero
Coral.calero@uclm.es
ALARCOS Research Group
Escuela Superior de Informatica, Universidad de Castilla-La Mancha
Ronda de Calatrava, 5, 13071 Ciudad Real, Spain

Michele Lanza
michele.lanza@unisi.ch
Faculty of Informatics
University of Lugano
Via G. Buffi 13, CH - 6900 Lugano, Switzerland

Geert Poels
geert.poels@ugent.be
Faculty of Economics and Business Administration - Ghent
University, and
Centre for Industrial Management – Katholieke Universiteit Leuven
Hoveniersberg 24, 9000 Gent, Belgium

Fernando Brito e Abreu
fba@di.fct.unl.pt
QUASAR Research Group
Faculdade de Ciências e Tecnologia, Universidade Nova de Lisboa
Monte da Caparica, Portugal

Contents

Code Analysis & Metrics

Quality assessment

Empirical studies

Investigating the Nesting Level of Composite States in UML Statechart Diagrams

José A. Cruz-Lemus[1], Marcela Genero[1], Mario Piattini[1] and Ambrosio Toval[2]

[1]ALARCOS Research Group
Department of Computer Science, University of Castilla – La Mancha
Paseo de la Universidad, 4 – 13071 Ciudad Real, Spain
{JoseAntonio.Cruz, Marcela.Genero, Mario.Piattini}@uclm.es

[2]Software Engineering Research Group
Department of Computer Science and Systems, University of Murcia
Campus de Espinardo – 30071 Murcia, Spain
atoval@um.es

Abstract. UML statechart diagrams are an important technique for modeling the dynamic aspects of software systems. We have been studying the relationship between many of the constructs of the UML statechart diagrams and the effect that they have on the understandability of the diagrams themselves, including composite states. We found that the use of composite states make UML statechart diagrams easier to understand. In order to go a step further, we decided to investigate if the nesting level of composites states (NLCS) has an impact on the understanding of the diagrams through a controlled experiment. Unfortunately, the obtained results are not quite conclusive and we have not been able to find an optimal use of nesting within UML statechart diagrams. So, further empirical research is needed.

1. Introduction

Modeling is at the core of many disciplines, but it is especially important in engineering because it facilitates the communication and construction of complex things from smaller parts [21]. New approaches in software engineering like MDA (Model Driven Architecture) [17] and MDD (Model Driven Development) [1] are enabling a shift in focus from software to models of software. These approaches consider models as end-products rather than just mean to produce software. For that reason, models like UML models are gaining more relevance in the development of software. The quality of the models used is of great importance as it will ultimately determine the quality of the software systems produced.

UML statechart diagrams have become an important technique for modeling the dynamic aspects of a software system and are also an important element of OO design documents [13].

In previous researches, we have studied the relationship between many of the constructs of the UML statechart diagrams and the effect that they have on the understandability of the diagrams themselves. To do so, we had previously defined and

validated, both theoretically and empirically, a set of metrics for evaluating the structural properties of UML statechart diagrams, such as size and complexity [11]. In all these works we had found that the usage of composites states had no influence on the understandability of UML statechart diagrams. This fact seemed to be a bit suspicious. For that reason, we carried out an experiment and its replica for studying specifically if the use of composite state facilitated or not the understanding of UML statechart diagrams [10]. The results of this empirical study indicated that the use of composite states improves the understandability efficiency of UML statechart diagrams if the subjects have a certain level of experience in working with this kind of UML diagrams. Thus, we concluded that using composite states when modeling the behaviour of systems through UML statechart diagrams makes them more understandable.

Based on these findings, we have defined a new metric named *Nesting Level in Composite States* (NLCS) which indicates the maximum number of composite states nesting in an UML statechart diagram.

In this work, we will try to empirically validate this metric, taking as basis the measure DIT (Depth of Inheritance Tree) defined in [8]. We think that there is a certain level of parallelism between the nesting level of composite states within an UML statechart diagram and the class inheritance in an UML class diagram.

Besides, we can base our research on the different experimental experiences that the metric DIT has been studied under [3-7, 9, 12, 14, 18, 19, 22]. We will use some of the results of these works in order to design and perform a controlled experiment and try to find out the optimal nesting level of composite states within an UML statechart diagram.

In section 2 we define our research question. Section 3 describes the experimental process, covering the design, tasks and performance of the experiment. Section 4 describes the data analysis and the interpretation of the obtained results. Finally, conclusions and future work are presented in section 5.

2. Research Question

As we commented in section 1, our research question can be stated as:

> *Does the use of different nesting levels of composite states within an UML statechart diagram affect the understandability of the diagrams?*

In order to answer this question we have defined the previously presented metric NLCS. Based on the guidelines exposed in [18], we have formulated the following experimental hypotheses:

- H_{0ij}: the understandability of UML statechart diagrams with i and j composite states nesting levels is not significantly different,
- H_{1ij}: the understandability of UML statechart diagrams with i and j composite states nesting levels is significantly different,

  in both cases, $i, j \in \{0, 1, 2\}$ and $i \neq j$.

This way, there are three different null hypotheses (H_{01}, H_{02}, H_{12}), taking account of symmetries (H_{12}=H_{21}).

3. Experimental Process

In this section, we describe the controlled experiment that we have carried out for testing the hypotheses stated in the previous section. All the experimental process is based on the guidelines outlined in [24].

The experiment took place at the University of Murcia (Spain) in May 2005 and its main features are detailed in the following sub-sections.

3.1. Subjects

38 subjects from the University of Murcia participated in this experiment. 11 of them were on their 4th year of Computer Science whilst the rest had finished their Computer Science studies less than one year before.

The tasks to be performed did not require high levels of industrial experience, so experiments with students could be considered as appropriate [2, 15]. Moreover, students are the next generation of people entering this profession, so they are close to the population under study [16]. Besides, working with students implies a set of advantages [23], such as the fact that the prior knowledge of the students is rather homogeneous. The availability of a large number of subjects is another plus point.

All the subjects had received a complete Software Engineering course in which they had studied modeling techniques, including UML. They also received a short training session before the performance of the experiment, in which the main constructs of UML statechart diagrams were commented on and where some examples of the tasks to be performed by them were explained by the conductor of the experiment. So we consider that the level of experience they brought to the experiment was acceptable.

3.2. Experimental Design

Our dependent variable was the understandability of UML statechart diagrams and we would measure this through three different ways:

- Effectiveness, i.e., number of correct answers vs. total number of asked questions.
- Efficiency, i.e., correct answers given by the subjects vs. time spent on answering the asked questions.

Our independent variable was the nesting level of the different UML statechart diagrams that were part of the experimental material. We used three diagrams with 0, 1 and 2 nesting levels respectively that modelled exactly the same system (an ATM) and were conceptually identical. These were also the values of the metric NLCS for each of the diagrams.

The original experimental material can be found in Appendix A at the end of the document.

3.3. Experimental Task

Each subject received one diagram out of the three possibilities. The Universe of Discourse (UoD) of the diagrams was quite usual and not exceptional at all, so that there was no need for extra effort in understanding them.

Each diagram had a test which contained 9 questions which were exactly the same for the three different options. The questions inquired about what state would be reached after the triggering of some events, which state would be reached after a certain sequence of events and guard conditions or what sequence was the minimum possible for going from one given state to another, for instance. The subjects had to note down the times at which they started and finished answering the whole questionnaire, as well as providing the answers to the questions themselves.

3.4. Experimental Procedure

The experiment started with a twenty-five-minute introductory session in which the conductor gave a general motivation for the experiment as well as the main elements of an UML statechart diagram. After that, the materials for the experiment were randomly distributed to the subjects.

In order to increase the motivation and interest of the subjects, the students were explained that the exercises that they were going to perform could be similar to those that would find in their exam at the end of the term.

At this point some examples in shortened version were performed by the conductor, who explained the correct answer to each question and the way of noting down the starting and finishing times properly.

Throughout this time, the subjects were allowed to ask the conductor about any doubt that they might have and to make any remarks they wished to.

4. Discussion of Results

All the data analysis presented in this section was carried out by means of SPSS [20].

4.1. Data Analysis

First, we carried out an analysis of the descriptive statistics of the data. The box-plots of the data shown in Figures 1 and 2 illustrate the statistics summarized in Table 1 and Table 2.

Table 1. Summary statistics for effectiveness

NLCS	Mean	Median	Min.	Max.	St. Dev.	Skew.	Kurtos.
0 (N=13)	0,820513	0,778	0,667	1	0,096635	0,8663	0,3516
1 (N=13)	0,790598	0,778	0,611	0,944	0,096635	-0,1927	-0,5104
2 (N=12)	0,736111	0,750	0,444	1	0,185206	-0,4030	-1,0046

Table 2. Summary statistics for efficiency

NLCS	Mean	Median	Min.	Max.	St. Dev.	Skew.	Kurtos.
0 (N=13)	0,014647	0,014675	0,009009	0,022388	0,003844	0,3130	-0,3562
1 (N=13)	0,013717	0,013722	0,006215	0,020151	0,004191	-0,2557	-0,8077
2 (N=12)	0,012510	0,012262	0,006203	0,018229	0,004149	-0,1652	-1,1057

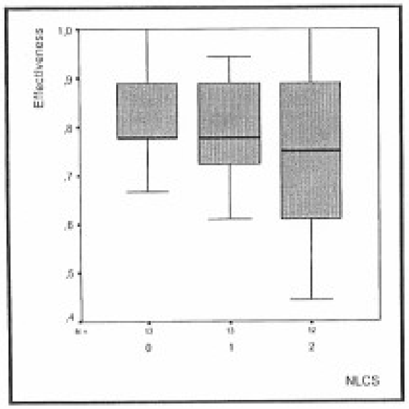

Fig. 1. Effectiveness box-plot

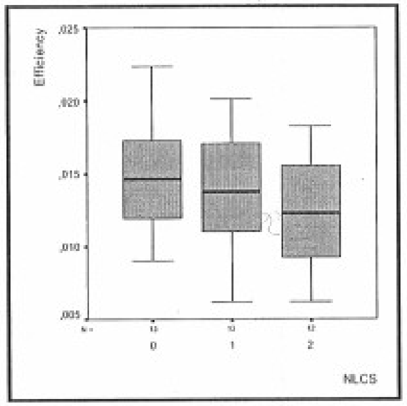

Fig. 2. Efficiency box-plot

In order to check the hypotheses presented in section 2, we performed some t-Tests with α=0.05. The obtained the results for the different dependent variables taking into account all the possible NLCS values are shown in Table 3.

Table 3. t-Tests results.

Dependent variable	NLCS	df	t Stat.	Sig.
Effectiveness	0 vs 1	24	0.789	0.438
	0 vs 2	16.281	1.411	0.177
	1 vs 2	16.281	0.911	0.376
Efficiency	0 vs 1	24	0.587	0.563
	0 vs 2	23	1.334	0.195
	1 vs 2	23	0.723	0.477

4.2. Results Interpretation

Both for effectiveness and efficiency, the mean values for 0 and 1 nesting levels are quite close, while the mean values for 2 nesting levels are much lower. In our opinion, this means that a flat nesting level (0 or 1 levels) helped the subject to a better understanding of the diagrams than a big nesting level.

The results of the t-Tests performed do not allow us to reject any of the null hypotheses that we presented in section 2, as all the significance levels are above 0.05.

Anyway, these results are preliminary and further experimentation is needed. In fact, we will soon perform a replication of this experiment in order to empirically confirm our findings.

4.3 Threats to Validity

We must keep in mind a number of validity issues that are typically related to experiments of this type.

First, the subjects were not professional modelers. Obviously, we would expect much better results if the subjects were more experienced. However, the limited difficulty of the tasks and the UoD used make the students become suitable experimental subjects, as they are much easier to work with than some others. Nevertheless, further replications of this experiment using people already working in this profession would be really interesting.

Secondly, the diagrams that have been used represent relatively simple models and it is possible that if real-projects data were used, we would obtain different results, although we contend that the conclusions reached would be the same as in this case.

5. Conclusions and Future Work

Worried about how UML constructs impact on the understandability of UML statechart diagrams we carried out several empirical studies. The results obtained in the last one [10] revealed that the use of composite states improves the understandability efficiency of UML statechart diagrams if the subjects have a certain level of experience in working with this kind of UML diagrams.

In this work, we have defined a new metric called Nesting Level in Composite States (NLCS) which indicates the maximum number of composite states nesting in an UML statechart diagram. In addition, we have detailed a controlled experiment that we have performed in order to investigate if the values of NLCS have an impact on the understanding of UML statechart diagrams, as none of the test performed to the data showed a significant difference between the understandability effectiveness and efficiency of the different diagrams uses, each one with a different value for NLCS.

The findings obtained through the experimentation were not really conclusive. We have not been able to find an optimal use of nesting within UML statechart diagrams, and we can only conclude that a flat nesting level (0 or 1) within an UML statechart diagram makes it more understandable.

In order to strengthen this opinion and try to reach some new and more conclusive results and conclusions, we have decided to perform a replication of the experiment presented in this work.

Acknowledgements

This research is part of the MESSENGER project (PCC-03-003-1) financed by 'Consejería de Ciencia y Tecnología de la Junta de Comunidades de Castilla-La Mancha (Spain)', and the DYNAMICA/PRESSURE and the DYNAMICA/CALIPO projects supported by 'Dirección General de Investigación del Ministerio de Ciencia y Tecnología (Spain)' (TIC2003-07804-C05-05)

References

1. Atkinson, C., Kühne, T.: Model Driven Development: a Metamodeling Foundation. IEEE Transactions on Software Engineering. 20 (2003) 36-41
2. Basili, V., Shull, F., Lanubile, F.: Building Knowledge through Families of Experiments. IEEE Transactions on Software Engineering. 25 (1999) 456-473
3. Briand, L., Morasca, S., Basili, V.: Property-Based Software Engineering Measurement. IEEE Transactions on Software Engineering. 22 (1) (1996) 68-86
4. Briand, L., Wüst, J., Lounis, H.: Investigating Quality Factors in Object-oriented Designs: An Industrial Case Study. Technical Report ISERN 98-29, version 2 (1998)
5. Briand, L., Wüst, J., Daly, J., Porter, V.: Exploring the Relationships between Design Measures and Software Quality in Object-Oriented Systems. The Journal of Systems and Software. 51 (2000) 245-273
6. Briand, L., Bunse, C., Daly, J.: A Controlled Experiment for Evaluating Quality Guidelines on the Maintainability of Object-Oriented Designs. IEEE Transactions on Software Engineering. 27 (6) (2001) 513-530
7. Cartwright, M.: An Empirical View of Inheritance. Information and Software Technology. 40 (4) (1998) 795-799
8. Chidamber, S., Kemerer, C.: A Metrics Suite for Object-Oriented Design. IEEE Transactions on Software Engineering. 20 (1994) 476-493
9. Chidamber, S., Darcy, D., Kemerer, C.: Managerial Use of Metrics for Object-Oriented Software: An Exploratory Analysis. IEEE Transactions on Software Engineering. 24 (8) (1998) 629-639
10. Cruz-Lemus, J. A., Genero, M., Manso, M. E., Piattini, M.: Evaluating the Effect of Composite States on the Understandability of UML Statechart Diagrams. Proc. of ACM/IEEE 8th International Conference on Model Driven Engineering Languages and Systems (MODELS / UML 2005). Montego Bay, Jamaica (2005) (under submission)
11. Cruz-Lemus, J. A., Genero, M., Piattini, M.: Metrics for UML Statechart Diagrams. In: Genero, M., Piattini, M., Calero, C. (eds.): Metrics for Software Conceptual Models. Imperial College Press, United Kingdom (2005)
12. Daly, J., Brooks, A., Miller, J., Roper, M., Wood, M.: An Empirical Study Evaluating Depth of Inheritance on Maintainability of Object-Oriented Software. Empirical Software Engineering. 1 (2) (1996) 109-132
13. Denger, C., Ciolkowski, M.: High Quality Statecharts through Tailored, Perspective-Based Inspections. Proc. of 29th EUROMICRO Conference "New Waves in System Architecture". Belek, Turkey (2003) 316-325
14. Harrison, R., Counsell, S., Nithi, R.: Experimental Assessment of the Effect of Inheritance on the Maintainability of Object-Oriented Systems. The Journal of Systems and Software. 52 (2000) 173-179
15. Höst, M., Regnell, B., Wohlin, C.: Using Students as Subjects - a Comparative Study of Students & Proffesionals in Lead-Time Impact Assessment. Proc. of 4th Conference on Empirical Assessment & Evaluation in Software Engineering (EASE 2000). Keele, UK (2000) 201-214
16. Kitchenham, B., Pfleeger, S., Pickard, L., Jones, P., Hoaglin, D., El-Emam, K., Rosenberg, J.: Preliminary Guidelines for Empirical Research in Software Engineering. IEEE Transactions on Software Engineering. 28 (8) (2002) 721-734
17. OMG: MDA - The OMG Model Driven Architecture. Object Management Group (2002)
18. Poels, G., Dedene, G.: Evaluating the Effect of Inheritance on the Modifiability of Object-Oriented Business Domain Models. Proc. of 5th European Conference on Software Maintenance and Reengineering (CSMR 2001). Lisbon, Portugal (2001) 20-29
19. Prechelt, L., Unger, B., Philippsen, M., Tichy, W.: A Controlled Experiment on Inheritance Depth as a Cost Factor for Code Maintenance. The Journal of Systems and Software. 65 (2003) 115-126
20. SPSS: SPSS 11.5, Syntax Reference Guide. SPSS Inc. Chicago, USA. (2002)
21. Thomas, D.: MDA: Revenge of the Modelers or UML Utopia? IEEE Software. 21 (3) (2004) 15-17
22. Unger, B., Prechelt, L.: The Impact of Inheritance Depth on Maintenance Tasks - Detailed Description and Evaluation of Two Experimental Replications. Technical Report, Karlsruhe University (1998)
23. Verelst, J.: The Influence of the Level of Abstraction on the Evovability of Conceptual Models of Information Systems. Proc. of 3rd International Syposium on Empirical Software Engineering (ISESE 2004). Redondo Beach, USA (2004) 17-26
24. Wohlin, C., Runeson, P., Hast, M., Ohlsson, M. C., Regnell, B., Wesslen, A.: Experimentation in Software Engineering: an Introduction. Kluwer Academic Publisher (2000)

Appendix A: Experimental Material

In this appendix we show the experimental material handed out to the subjects in the experiment. The three original (Spanish) diagrams model the behaviour of an ATM machine. The first one does not use composite states, while in the second and third diagrams the nesting level value is 1 and 2 respectively.

After them, we can find a translated version of the questionnaire attached to the diagrams.

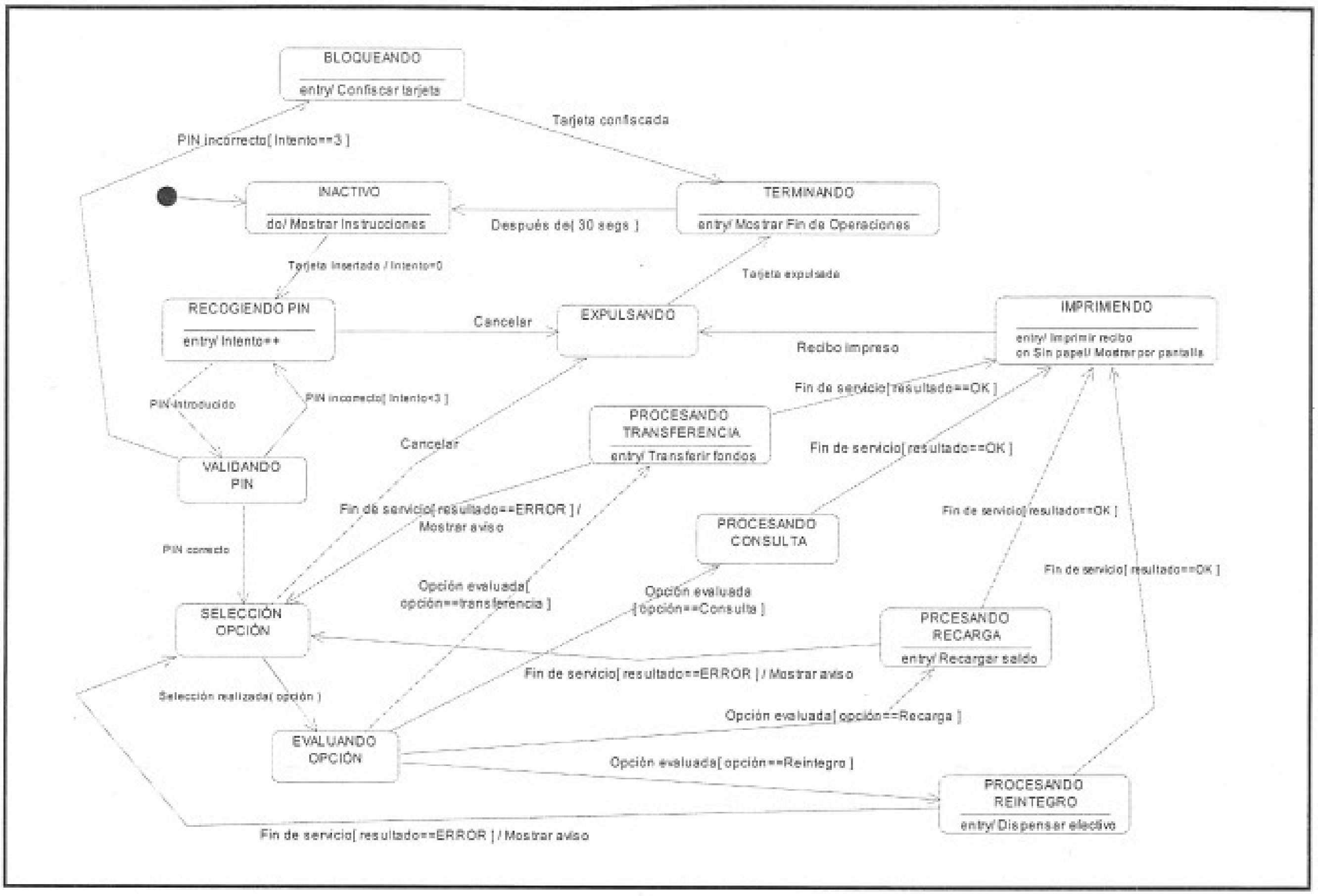
BLOQUEANDO
entry/ Confiscar tarjeta
Tarjeta confiscada
PIN incorrecto[Intento==3]
INACTIVO
do/ Mostrar Instrucciones
Después de(30 segs)
TERMINANDO
entry/ Mostrar Fin de Operaciones
Tarjeta insertada / Intento=0
Tarjeta expulsada
RECOGIENDO PIN
entry/ Intento++
Cancelar
EXPULSANDO
Recibo impreso
IMPRIMIENDO
entry/ Imprimir recibo
on Sin papel/ Mostrar por pantalla
PIN introducido
PIN incorrecto[Intento<3]
Fin de servicio[resultado==OK]
PROCESANDO TRANSFERENCIA
entry/ Transferir fondos
Cancelar
VALIDANDO PIN
Fin de servicio[resultado==OK]
Fin de servicio[resultado==ERROR] / Mostrar aviso
PROCESANDO CONSULTA
Fin de servicio[resultado==OK]
PIN correcto
Fin de servicio[resultado==OK]
Opción evaluada[opción==transferencia]
Opción evaluada [opción==Consulta]
SELECCIÓN OPCIÓN
PRCESANDO RECARGA
entry/ Recargar saldo
Fin de servicio[resultado==ERROR] / Mostrar aviso
Selección realizada(opción)
Opción evaluada[opción==Recarga]
EVALUANDO OPCIÓN
Opción evaluada[opción==Reintegro]
PROCESANDO REINTEGRO
entry/ Dispensar efectivo
Fin de servicio[resultado==ERROR] / Mostrar aviso

Fig. 3. Experimental material (NLCS=0)

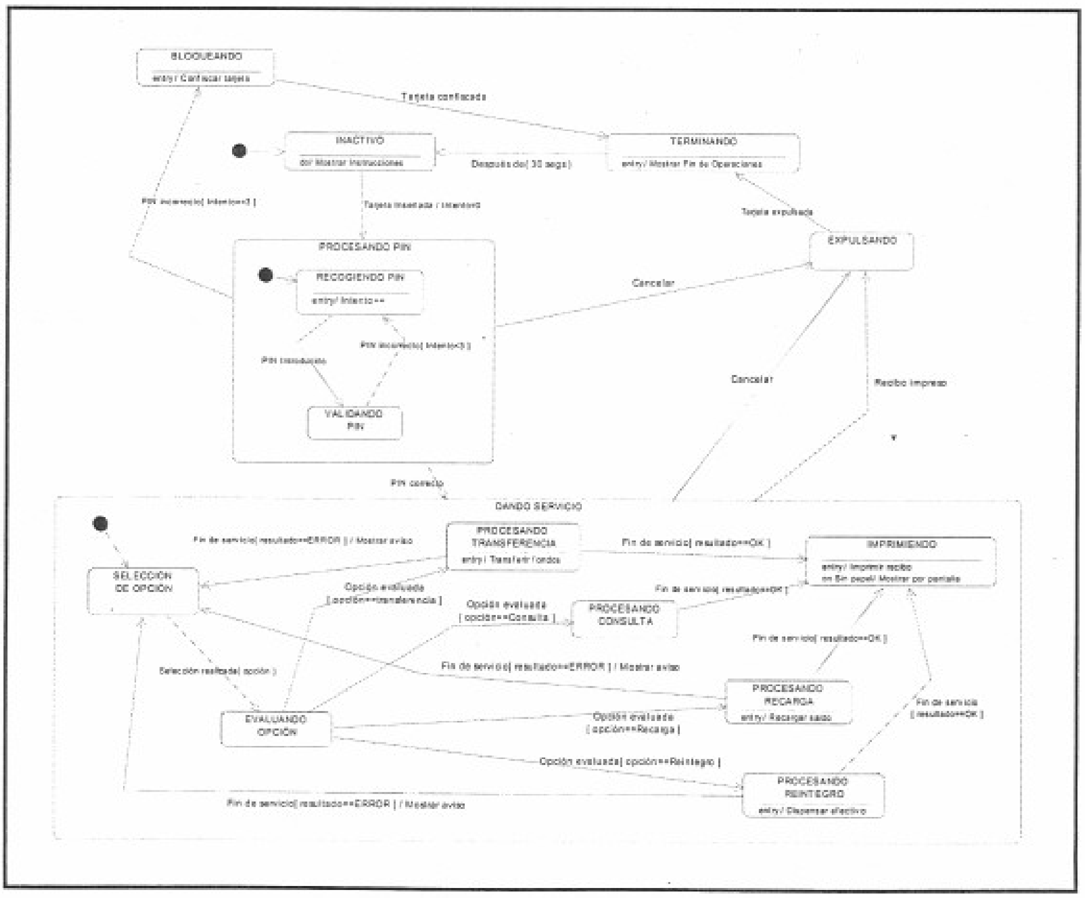
BLOQUEANDO
entry/ Confiscar tarjeta
Tarjeta confiscada
INACTIVO
do/ Mostrar Instrucciones
Después de(30 segs)
TERMINANDO
entry/ Mostrar Fin de Operaciones
PIN incorrecto[Intento==3]
Tarjeta insertada / Intento=0
Tarjeta expulsada
PROCESANDO PIN
EXPULSANDO
RECOGIENDO PIN
entry/ Intento++
Cancelar
PIN incorrecto[Intento<3]
PIN introducido
Cancelar
Recibo impreso
VALIDANDO PIN
PIN correcto
DANDO SERVICIO
Fin de servicio[resultado==ERROR] / Mostrar aviso
PROCESANDO TRANSFERENCIA
entry/ Transferir fondos
Fin de servicio[resultado==OK]
IMPRIMIENDO
entry/ Imprimir recibo
on Sin papel/ Mostrar por pantalla
SELECCIÓN DE OPCIÓN
Opción evaluada [opción==transferencia]
Opción evaluada [opción==Consulta]
Fin de servicio[resultado==OK]
PROCESANDO CONSULTA
Fin de servicio[resultado==OK]
Selección realizada(opción)
Fin de servicio[resultado==ERROR] / Mostrar aviso
PROCESANDO RECARGA
entry/ Recargar saldo
Fin de servicio [resultado==OK]
EVALUANDO OPCIÓN
Opción evaluada [opción==Recarga]
Opción evaluada[opción==Reintegro]
PROCESANDO REINTEGRO
entry/ Dispensar efectivo
Fin de servicio[resultado==ERROR] / Mostrar aviso

Fig. 4. Experimental material (NLCS=1)

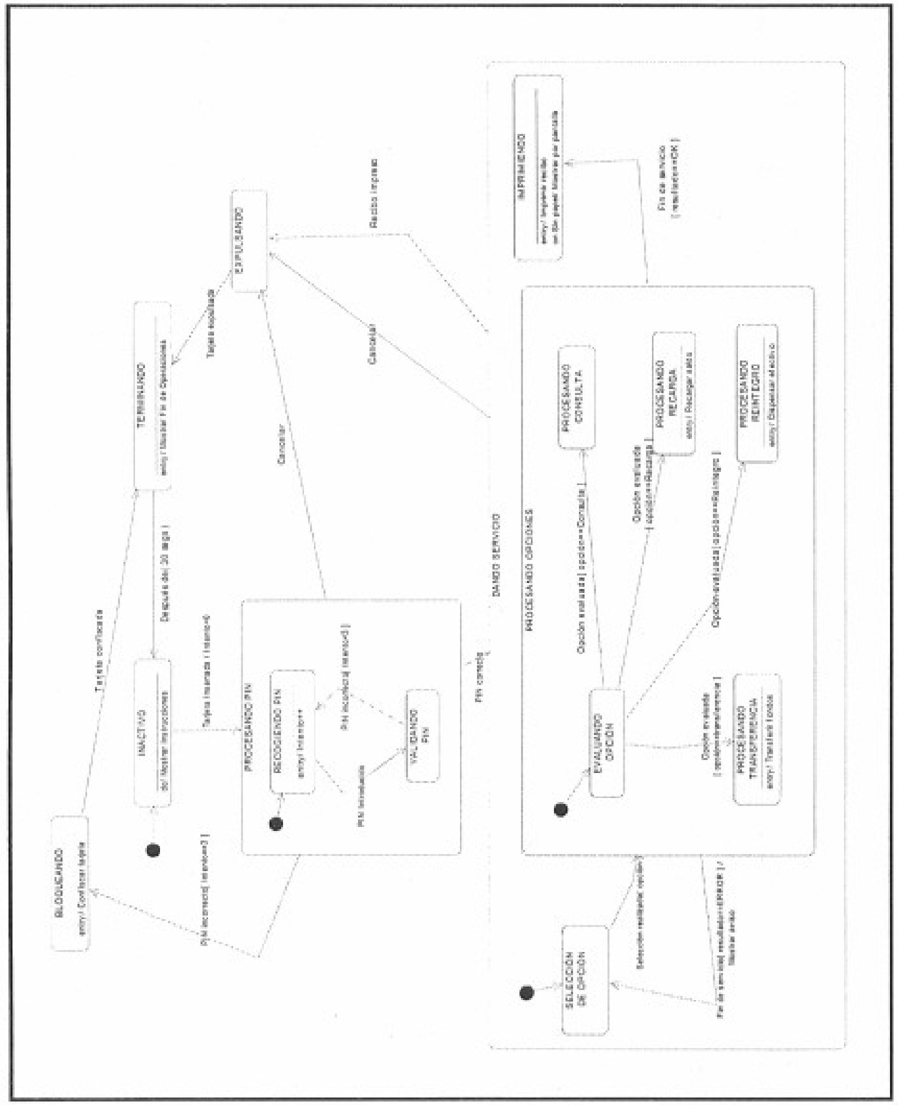

Fig. 5. Experimental material (NLCS=2)

CHECK TIME (HH : MM : SS) __ : __ : __

1. If we are in the state IMPRIMIENDO and the event *Recibo impreso* occurs, which state do we reach?

2. If while being in the state SELECCIÓN DE OPCIÓN the event *Selección realizada* occurs and the variable opción has the velue Consulta, which state do we reach?

3. Which state do we reach if while being in the state INACTIVO the following sequence of events occurs?
   a. Tarjeta insertada
   b. Pin introducido
   c. Pin incorrecto
   d. Pin introducido
   e. Pin correcto

4. Which is the **minimum** sequence of events and guard concitions necessary for going from the state SELECCIÓN DE OPCIÓN to the state INACTIVO?

5. Which is the value of the variable Intento if starting from the state INACTIVO the following sequence of events occurs?
   a. Tarjeta insertada
   b. Pin incorrecto
   c. Pin correcto

6. If we are in the state SELECCIÓN DE OPCIÓN and the event *Cancelar* occurs, which state do we reach?

7. If while being in the state PROCESANDO REINTERGRO the event *Fin de servicio* occurs and the variable resultado has the velue ERROR, which state do we reach?

8. Which state do we reach if while being in the state SELECCIÓN DE OPCIÓN the following sequence of events occurs?
   a. Selección realizada
   b. Opción evaluada
   c. Fin de servicio
   d. Recibo impreso

9. Which is the **minimum** sequence of events and guard concitions necessary for going from the state INACTIVO to the state TERMINANDO?

CHECK TIME (HH : MM : SS) __ : __ : __