Application of the Relational Concept Analysis (RCA) for Remodularization of a Software Architecture and Comparison with Others Techniques Based on FCA and Oriented Graph

Lala Madiha Hakik†, Rachid El Harti††

† Faculty of Science and Techniques, University Hassan I, BP 577, Settat, Morocco.
†† Current address: Umm Al Qura University, College of applied sciences. Mekkah. Kingdom of Saudi Arabia, KSA.
†† Permanent address: Faculty of Science and Techniques, University Hassan I, BP 577, Settat, Morocco.

Summary
In a previous study we proceeded to the remodularization architecture based on classes and packages using the Formal Concept Analysis (FCA) [2] [13] [14], we then got two possible remodularized architectures and we explored the issue of redistributing classes of a package to other packages, we used an approach based on Oriented Graph to determine the packages that receive the redistributed classes and we evaluated the quality of a remodularized software architecture by metrics[1]. This paper presents the usefulness of relational concept analysis (RCA) for remodularization of a software architecture composed of classes and packages and we evaluate the quality of the result by metrics of coupling and cohesion. Also we compare results obtained by application of techniques based on Relational concept analysis (RCA) between Formal Concept analysis and Oriented graph.
Keyword*: Remodularization, Software architecture, Relational Concept Analysis (RCA), Metrics of Coupling and Cohesion.*

1. Introduction

Great software systems based on approaches, the object consist of classes grouped into packages, forming a modular structure. The dependency relationships between classes in the same package (internal dependencies), and between classes of different packages (external dependencies generate complexity making it difficult to understand and maintain the system. In addition, the modular structure tends to degrade over time, making necessary an expert intervention for modernization [1].
Many researchers make proposals on this subject using technical visualization, algorithms of remodularization, Exploring the Redistribution Classes of a Package with an Approach Based on Formal Concept Analysis. [13] [14] or using an approach based on Oriented Graph based on the technique of shortest path [1] .
In this paper, we study a particular declination, cf. the problem presented by H. Abdeen et al. [2] [1], which is about the redistribution of classes from one system to existing packages. Namely, we consider in this paper more precisely the redistribution of classes in a package to other packages[1].
This package may be a very small and in fact we want to balance the sizes of packages in the system, or it was artificially created to contain added classes to the system and the designer considers that there is no consistency semantics[1].
We explore a solution using *relational concept analysis (RCA)* and illustrate our proposal with a theoretical example.
Section 2 presents our example, then we describe the approach in Section 3. Section 4 presents validation metrics of cohesion and coupling measure and we discuss our main results. The comparison of results obtained by application of techniques based on Relational concept analysis (RCA) between Formal Concept analysis and Oriented graph in Section 5. Related work is presented in Section 6, and then we conclude in Section 7.

2. Illustration

This section presents the problem of software architectures remodularization on an example. We will use the architecture shown in Figure 1 consists of five packages A, B, C, D and E. Packages A, B, C, D, E are expected to contain more classes that are not shown for simplicity. Dependencies linking classes: they correspond for example to call a method or use of a type. External dependency relationships link classes of package E to classes of other packages. Internal dependency relationships connect classes E between. Internal dependencies of A, B, C and D are not presented.
We are interested in the redistribution of classes E to other packages with an exploratory method, whose proposals for redistribution are then presented to an expert. These proposals are based on the idea that the expert, while checking the semantic classes, could search for the increase of the cohesion (within the meaning of the coupling of classes in a package) and reduce the coupling between classes in different packages. To do this, we

Manuscript received March 5, 2014
Manuscript revised March 20, 2014

believe it is appropriate to encourage the following two trends [12][13]:

- Classes in a package attract them to classes of E,
- If classes of E are interconnected, it is better to redistribute in the same package.

We believe that the Relational Concept Analysis (RCA) can bring interesting ways to solve this problem because this technical method allows the group to connect classes identically. We are looking to propose a solution to an expert.

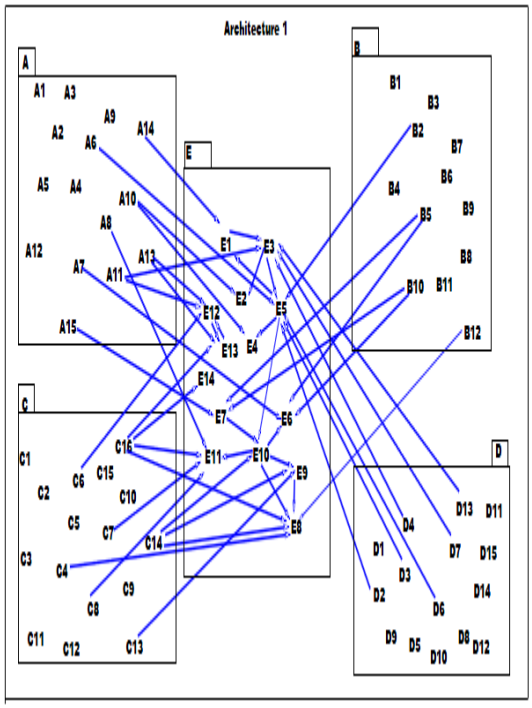

Figure 1. An initial architecture composed of classes and packages [12][13].

3. Proposed approach

The Relational Concept Analysis (RCA[1]) is a technical relational data analysis whose objects are described by attributes and relations with other objects. The RCA is used in software engineering for solving several problems. For redistributing of classes of package E to the packages A, B, C, D of the software architecture for remodularization, our exploration was carried following the steps below:

Step1: Relational context family[1]
It's a simple entity relationship mode to introduce RCA.

[1]http://www.lgi2p.ema.fr/~urtado/Slides/Huchard_partie1_14_02_2013.pdf

- **Object- attribute contexts:**

For our case, we have only the object- attribute contexts are used to build the foundation of the concept lattice family (see figure 4) result of grouping two lattices T(C1) (figure 2) and T(C2) (figure 3).
Configurations In the context of our problem, we studied six different configurations.
We present two of them.
The configuration with RCA is to define a formal context C: the set O of entities studied (or formal objects) Set A of characteristics (or formal attributes) and the relationship
$R \subseteq O \times A$.

The first formal context associates a class c of a package E to the packages that access to this class c (see Figure 2, left panel).
Context (formal context C1).
- O1 is the set of classes of E in relation to the outside.
- A1 is the set of packages A, B, C, D (which has a relation to a class of E).
- R1 is the relation "is a target for external access".
- (e, p) ∈ R1 if e is an access target from p, for example (E2, A) ∈ R1.

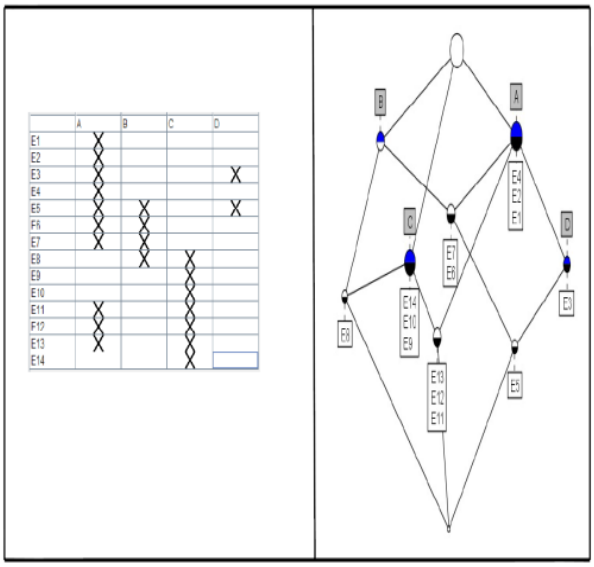

Figure 2. Formal context C1 and lattice T(C1) –Architecture 1-[12][13].

The second formal context can refine the results and redistribute the same package into two classes that are interconnected in E). It combines a class of package E another class that is connected (see Figure 3, left panel).

Context (formal context C5).

- O5 is the set of classes of E in relation to the outside.
- A5 = O5: E classes in relation to the outside.
- R5 is the relation "is connected to".
- (e1, e2) ∈ R5 if there is an arrow e1 to e2 or e1 to e2, for example (E4, E5) and (E5, E4) belong to R2.

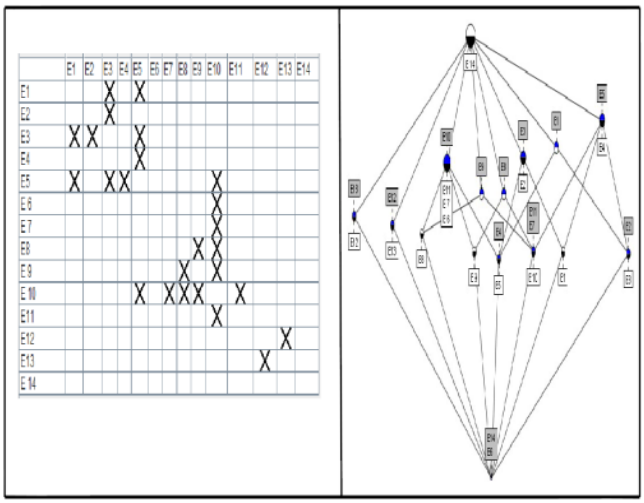

Figure 3. Formal context C2 and lattice T(C2) –Architecture 1-

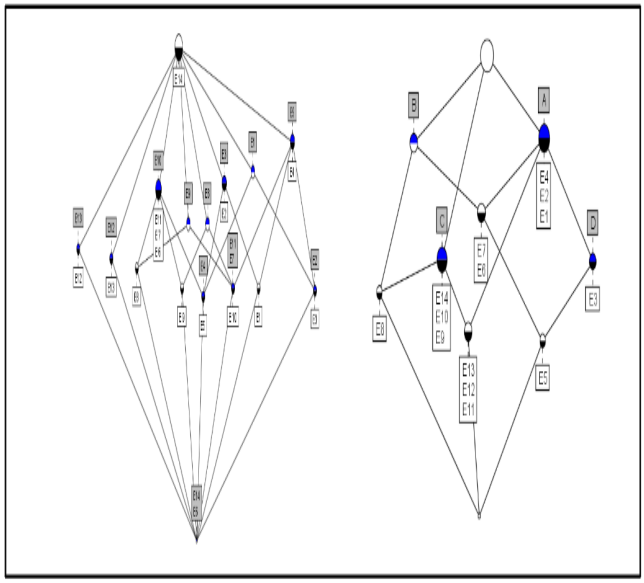

Figure 4. Concept lattice family grouping two lattices T(C1) (figure 2) and T(C2) (figure 3).

Step 2: RCA. Introducing relations as relational attributes

Relations between classes interconnected of package E and the target packages (see figure5), allow us in an exploration for the redistribution of classes of package E to know the exact destination of their specific assignment and gives rise to a new context producing a new concept lattice showing the execution of the redistribution of classes of package E(see figure 6).

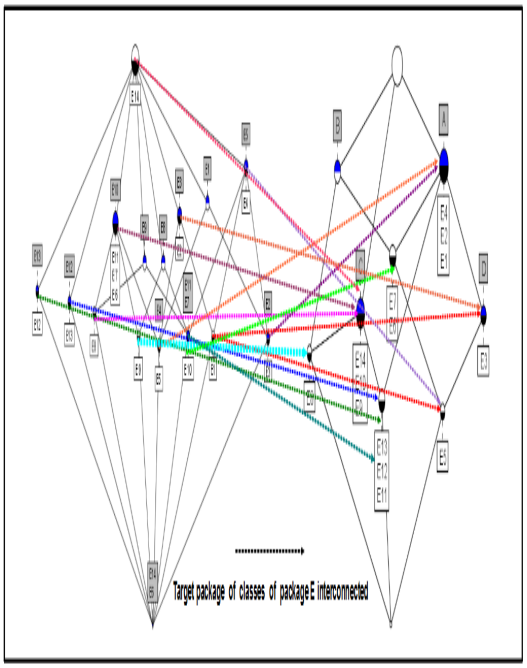

Figure 5. Relational concept family with relation **:**Target package of classes of package E interconnected

Step 3: RCA. Enriching relations

The relationship enrichment object of step 2 is done by replacing the objects columns by concepts lattice associated with the target context; the relationship is established by an operator of scaling[1] (see figure 6).

The New formal context associates a class c of a package E interconnected to the packages that access to this class c (see Figure 6, left panel).

New context (formal context C3).
- O3 is the set of classes of E in relation of the set of interconnected classes of E.
- A3 is the concept of T(C1) in relation of the set of interconnected classes of E.
- R3 is the relation "is a target package of classes of package E interconnected".

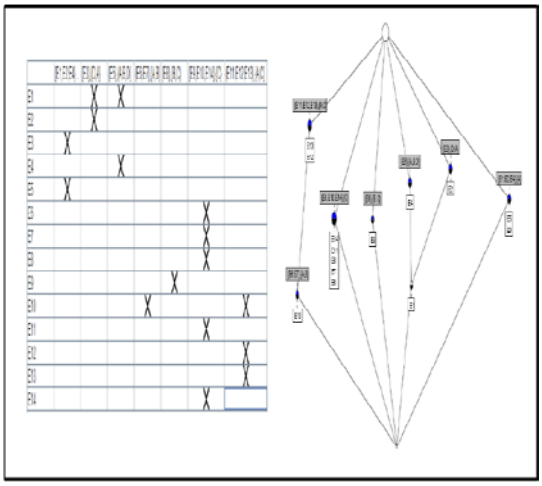

Figure 6. New Formal context C3 and new lattice T(C3) result of the relationship enrichment.

The concept lattice is the classification structures that expose concepts (their nodes) and link by specialization.
For example, the concept lattice T(C1) associated with context C1 (see Figure 2, right), contains eight concepts outside the top and bottom. The shaded part of the labels (upper part) corresponds to the simple intension of the concept, while the white portion of the label (lower part) is a simplified extension. Labeled extensions are inherited backwards in the lattice while labels intensions are inherited in descending.
For example the lattice T (C1) contains the concepts:
- ({E6, E7, E8}, {B}) at the top left, simplified in ({}, {B})
- ({E11, E12, E13}, {A, C}) in the middle at the bottom, simplified ({E11, E12, E13},{})

Example of exploration The exploration is to navigate the lattice T (C3) to identify opportunities for redistribution of classes and submit to an expert. We partially detail an example of analysis to explain the principle.
The lattice T (C3) can be divided into three large blocks in which we will choose concepts.

1. Analysis of the concept ({E5, E7}, {{E1, E2, E4}, {A}}) the right of T (C3): the expert can choose to put five classes E1, E2, E4, E5, E7 in A.
2. Analysis of the concept ({E6, E7, E8, E11, E14} {{E9, E10, E14}, {C}}) the left of T (C3): eight classes are in full extension of the concept of intension {C}, the expert can still choose to put them in C. The subsystem {E6, E7, E8, E11, E14, E9, E10, E14} can be put into C. Here we note that the class E7 can be assigned to package A or C and since it is interconnected to four classes that go to A and interconnected to eight classes for C, so it will go to the dominant package C.
3. Analysis of the concept ({E2}, {{E3}, {A, D}}) the right of T (C3): two classes are in full extension of the concept of intension { A, D}, the expert can choose to put two classes E2, E3 in package A or D and since E2 is interconnected to four classes that go to A and interconnected to one classe E3 that go to A or D, so E2 will go to the dominant package A. In this case the class E3 follow the class E2 also in package A to stay together.
4. Analysis of the concept ({E12, E13}, {{E11, E12, E13}, {A, C}}) the left of T (C3): three classes are in full extension of the concept of intension { A, C}, the expert can choose to put three classes E11, E12, E13 in package A or C and since E11 is interconnected to seven classes that go to C and interconnected to two classes E12 and E13 that go to A or C, so E11 will go to the dominant package C. In this case the classes E12 and E13 follow the class E11 also in package C to stay together.

Figure 7 shows one possible result of remodularized software architecture. The classes of package E deleted were distributed.

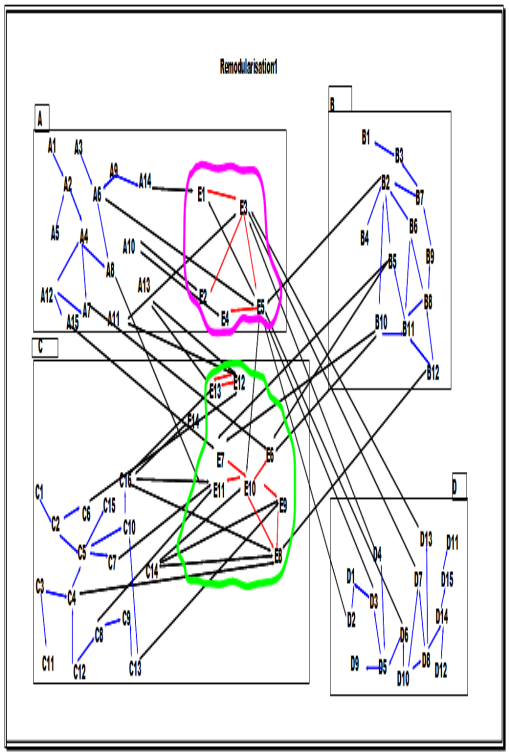

Figure 7. one possible result of remodularized software -architecture -1-

4. Results and discuss

4.1 Validation metrics

For validatiton of metrics cohesion and coupling, our calculations were based on figures 1 and 7 with an architecture comprising 5 packages A, B, C, D and E by redistribution classes of package E (figure 1) using The Relational Concept Analysis (RCA[1]), which resulted one possible remodularized architecture (figure7). The package E is removed during this operation.

1.Cohesion metics

Table 1. Cohesion metrics: Index of Package Goal Focus and Index of Package Services Cohesion.

	PF	IPSC
Package E of the original architecture 1	**0,5**	**0,0116**
Package A of the original architecture 1	**0**	**1**
Package C of the original architecture 1	**0**	**1**
Package A of the remodularization 1	**0,25**	**1**
Package C of the remodularization 1	**0,46**	**1**

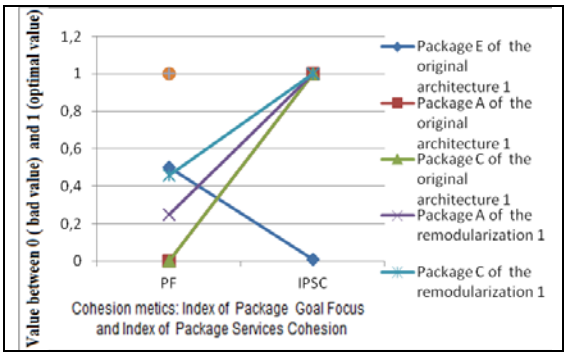

Figure 8. graphic representation of Cohesion metrics: Index of Package Goal Focus PF and Index of Package Services Cohesion IPSC (table1).

The Cohesion metrics: Index of Package Goal Focus (**PF**) and Index of Package Services Cohesion (**IPSC**) take their values from 0 to 1, where 1 is the optimal value and 0 is the wrong value.

Figure 8 gives the values of indices PF and IPSC for:

- Package E of Original Architecture 1 whose indexes are bad values because they are lower than 1.
- Packages A and C of remodularized architecture whose the index IPSC is optimal value 1 therefore very good.

4. 2 Coupling metrics

Table 2. Coupling metrics: Index of Inter-Package Interaction (IIPU and IIPE)

	IIPU	IIPE
The original architecture 1	**0,588**	**0,333**
Architecture of the remodularization 1	**0,811**	**1**

The coupling metrics: Index of Inter-Package Interaction (**IIPU** and **IIPE**) object of the figure 9, it is observed an improvement indexes IIPU and IIPE at remodularization 1 architecture compared to indexes of the original architecture 1 therefore a trend to optimality .

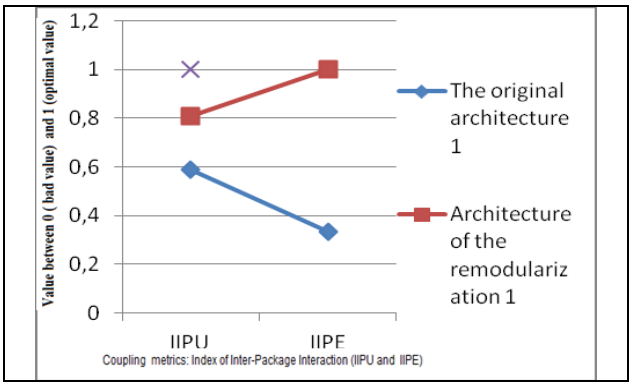

Figure 9. graphic representation of Coupling metrics: Index of Inter-Package IIPU and IIPE (table2).

Table 3. Coupling metrics: Index of Package changing Impact IPCI; Index of Package Communications Diversion (IIPUD and IIPED)

	IPCI	IIPUD	IIPED
Package E of the original architecture 1	**0**	**0,271**	**1**
Package A of the original architecture 1	**1**	**1**	**1**
Package B of the original architecture 1	**1**	**1**	**1**
Package C of the original architecture 1	**1**	**1**	**1**
Package D of the original architecture 1	**1**	**1**	**1**
The original architecture 1	**0,8**	**0,854**	**1**
Package A of the remodularization 1	**0**	**0,38**	**0,38**
Package B of the remodularization 1	**1**	**0,583**	**0,583**
Package C of the remodularization 1	**1**	**0,541**	**0,541**
Package D of the remodularization 1	**1**	**1**	**1**
Remodularization 1	**0,75**	**0,626**	**0,626**

Concerning the coupling metrics: Index of Package changing Impact (**IPCI**) and Index of Package

Communications Diversion (**IIPUD** and **IIPED**) presented in figure 10, the results obtained for remodularization 1 approximate from those of the original architecture 1 extend to a higher interesting value 0.626.

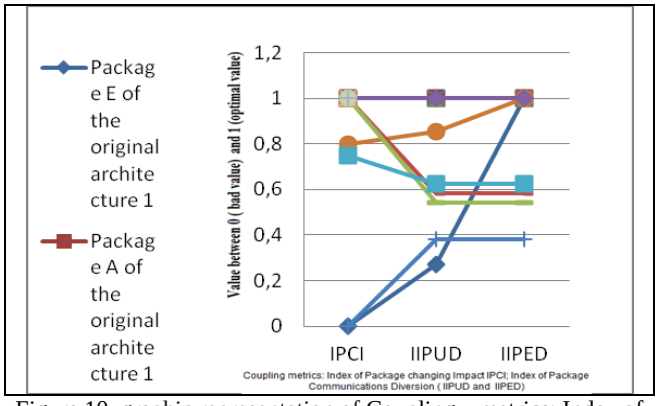

Figure 10. graphic representation of Coupling metrics: Index of Package changing Impact; Index of Package Communications Diversion (table3).

The results obtained at the level of the cohesion for the remodularized architecture 1 provides an optimum value 1 compared to the original architecture 1.
The results of the coupling have an improvement at the level of remodularized architecture 1 compared to the original architecture 1.

5. The comparison of results obtained by application of techniques based on Relational concept analysis (RCA) between Formal Concept analysis and Oriented graph

5.1. Comparison of remodularized software architectures obtained

Concerning the technical of redistribution of classes based on formal concept analysis, we got two remodularized software architecture offering an alternative choice to a software expert on one hand and know the way back to the original architecture on the other hand [2] [13] [14]
As to the result of the redistribution of classes in a package to other package by using the graph-oriented, this technique has generated one and unique remodularized software architecture [1].
Also the method of Relational concept analysis (RCA) generated one unique remodularized software architecture .

5.2. Comparison the results of the validation metrics coupling and cohesion

As a reminder, for validatiton of metrics cohesion and coupling, our calculations were based on figures 1 and 2 with an architecture comprising 5 packages A, B, C, D and E by redistribution classes of package E (using formal concept analysis techniques which resulted into two possible architectures . The package E is removed during this operation. Initial architecture (figure 1) and the two architectures result from the remodularization obtained by applying our approach based on formal concept analysis, which has been the object of the articles [2] [13][14] [29].
The results obtained at the level of the cohesion for the remodularization 1 and 2 provides an optimum value (with an advantage to the remodularization 1 remaining more performance for choosing a software expert). The results of the coupling have an improvement at the level of remodularized architectures 1 and 2 compared to the original architecture 1 [2] [29].
Furthermore the results obtained of the redistribution of classes in a package to other package by using the graph-oriented [1] [29], at the level of the cohesion for the remodularized architecture 1 provides an optimum value 1 compared to the original architecture 1. The results of the coupling have an improvement at the level of remodularized architecture 1 compared to the original architecture 1 [29] .
Also the method of Relational concept analysis (RCA) generated one unique remodularized software architecture and have revealed interesting results of the cohesion and coupling at the level of remodularized architecture 1 compared to the original architecture 1.
So the three techniques adopted for the redistribution of classes have revealed interesting results tending to optimization and limiting the number of remodularized software architectures proposed to the software expert .

6. Related Work

Different automated approaches have been proposed to restructure object systems. We cite three: the clustering algorithms, algorithms based on meta -heuristics and those based on the FCA[6]. The first aim to restructure system by the distribution of some elements (eg classes, methods , attributes) in groups such that the elements of a group are more similar to each other with elements of other groups [3] [7] [5]. Approaches to restructuring based on meta-heuristic algorithms [9] [8] are generally iterative stochastic algorithms, progressing towards a global

optimum of a function by evaluating a certain objective function (eg characteristics or quality metrics). Finally, the approaches based on FCA [10] [12] provide an algebraic derivation of hierarchies of abstractions from all entities of a system. Reference [4] presents a general approach for the application of the FCA in the field of object-oriented software reengineering. In previous work, we added the dimension of exploration using the FCA[13] [14].

Last work we explored the issue of redistributing classes of a package to other packages. We use an approach based on Oriented Graph to determine the packages that receive the redistributed classes and we have evaluate the quality of a remodularized Software Architecture by metrics for measuring Coupling and Cohesion of a Package[1] .

In this paper use an approach based on relational concept analysis (RCA) for remodularization of a software architecture composed of classes and packages and we evaluate the quality of the result by metrics of coupling and cohesion.

A large part of previous works related to oriented software metrics has focused on the issue of characterizing the class design, either looking internal complexity or relationship between a given class and other classes [1] [16] [17] [18] [19] [20] [21] [22] [23] [24] [25] [26].

In the literature, there is also a body of work that focus on object oriented metrics from the standpoint of their correlation with software changeability [16][27], or from the standpoint of their ability to predicate softwair maintenability [1] [16][28]. Other reasearchers argue that the measures resulted by the cohesion and coupling metrics of the previous works are open to interpretation [1] [16] [28].

In general, there are few metrics in the the literature devoted to packages.

Our cohesion and coupling metrics we provide in this work are similar to the metrics provided by Ducasse[1] [16].

Conclusion

In this article we explore the issue of redistributing classes of a package to other packages. We use an approach based on relational concept analysis (RCA) to determine the packages that receive the redistributed classes, and we have evaluate the quality of a remodularized software architecture by metrics for measuring coupling and cohesion of a package . The results have an improvement at the level of remodularized architecture.

we compare results obtained by application of techniques based on Relational concept analysis (RCA) between *Formal Concept analysis and Oriented graph.* So the three techniques adopted for the redistribution of classes have revealed interesting results tending to optimization and limiting the number of remodularized software architectures proposed to the software expert [29].

References

[1] L.M. Hakik, R. El harti. Technique Of Redistribution Classes Of A Package With An Approach Based On Oriented Graph And Evaluation Quality of A Remodularized Software Architecture. International Journal of Innovative Research in Science, Engineering and Technology.ISSN:2319-8753. Vol. 3, Issue 1, January 2014.

[2] L.M. Hakik, R. El Harti. Measuring Coupling and Cohesion to Evaluate the Quality of a Remodularized Software Architecture Result of an Approach Based on Formal Concept Analysis. IJCSNS International Journal of Computer Science and Network Security .Vol. 14 No. 1 pp. 11-16. Journal ISSN : 1738-7906. January 2014..

[3] F.B. Abreu, G. Pereira, and P. Sousa. A coupling-guided cluster analysis approach to reengineer the modularity of object-oriented systems. In Proceeding of the confeence on Software Maintenance and Reengineering. CSMR ‘OO, pages 13-, Washington, DC, USA, 2000. IEEE Compter Society Press.

[4] G. Arévalo, S. Ducass, and O. Nierstrasz. Lessons leaned in appling fomal concept analysis to reverse engineering. In Proceeding of the Third international conference on Fomal Concept Analysis, ICFCA’05, pages 95-112, Berlin. Heidelberg, 2005. Spinge-Velag.

[5] M. Bauer and M. Trifu. Architecture-aware adaptive clustering of oo s ystems. In Poceedings of the Eighth Euromicro Working Conference on Software Maintenance and Reengineering (CSMR ‘O4), CSMR ‘O4, pages 3-, Washington, DC, USA, 2004. IEEE Compter Society.

[6] B. Ganter and R. Wille. Formal Concept Analysis. Mathematical Fondations. Spinge. 1999.

[7] B.S. Mitchell and S. Mancoridis. Compaing the decompositions produced by software clustering algoithms using similarity measurements. In ICSM, pages 744-753, 2001.

[8] M.O’Keeffe and M. i Cinneide. Seach-based refactoring fo software maintenance. J. Syst. Softw., 81(4): 502-216, April 2008.

[9] O. Seng, J. Stammel and D. Burkhart. Search- based determination of refactorings for improving the class structure of object-oriented systems, In Mike Cattolico, edito. GECCO, pages 1909-1916. ACM, 2006.

[10] G.Snelting. Software reengineering based on concept lattices. In CSMR, pages 3-10, 2000.

[11] T. Tilley, R. Cole, P. Becker, P.W. Eklund. A survey of formal concept analysis support for software engineering activities. In Int. Conf. Fomal Concept Analysis (ICFCA 2005), pages 250-271, 2005.

[12] P. Tonella.Concept analysis for module restructuring. IEEE Trans. Software Eng..27 (4): 351-363, 2001.

[13] L.M. Hakik, M. Huchard, R. El Harti et A.D. Seriai. Exploration de la redistribution des classes d'un package par des techniques d'Analyse Formelle de Concepts. The first conference in software engineering (CIEL 2012), France, 2012.

[14] L.M. Hakik, R. El Harti . "*Exploring the Redistribution Classes of a Package with an Approach Based on Formal Concept Analysis* ", Vol.2 - Issue 12 (December - 2013), International Journal of Engineering Research &

Technology (IJERT), ISSN: 2278-0181, www.ijert.org.
[15] A. Anwar. Formalisation par une approche IDM de la composition de modeles dans le profil V*UML*. Thesis. Toulouse *Univ*ersity. *2009*.
[16] S. Ducasse, N. Anquetil, M.U. Bhatti and A.C. Hora. Software metrics for package remodularisation. Research report, November 2011.
[17] S. R. Chidamber and C. F .Kemer. A metrics suit for object oriented design. IEEETSE, 20: 476-493, 1994.
[18] F.B. Abreu and R. Carapuca. Candidate metrics for objected-oriented software within a taxonomy framework. Journal of Sys, Sof. 26: 87-96, 1994.
[19] W. Li and S. Henry. Objected-oriented metrics that predict maintainability. Journal of Sys, Sof. 23:111-112, 1993.
[20] W. Li. Another metric suit for object oriented programming. Journal of Sys, Sof. 44:155-162, 1998.
[21] B.H. Selers. Object-Oriented Metrics: Measures of Complexity. Prentice-Hall, 1996.
[22] J.M. Bieman and B.K. Kang. Cohesion and reuse in an object-oriented system. In ACM Symposium on Software Reusability. April 1995.
[23] J.M. Bieman and B.K. Kang. Measuring design-level cohesion. IEEETSE, 24(2) :111-124, February 1998.
[24] L.C. Briand, S. Morasca and V. R. Basili. Defining and validation measures for object-based high-level design. IEEE TSE, pages : 722-743, 1999.
[25] L.C. Briand, J.W Daly and J. Wust. A Unified Framework for Cohesion Measurement in Objected-Oriented Systems. Empirical Software Engineering. An International Journal, 3(1):65-117, 1998.
[26] L.C. Briand, J.W Daly and J. Wust. A Unified Framework for Coupling Measurement in Objected-Oriented Systems. IEEETSE, 25(1):91-121, 1999.
[27] R.K. Bandi, V.K. Vaishnavi and D.E. Tuk. Predicting maintenance performance using object- oriented design complexity *met*rics. IEEETSE, 29: 77-87, 2003.
[28] H. Kabaili, R.K. Keller, *F.* Lustman. Cohesion as changeability *indicato*r in object- oriented systems. In Fifth Europ. Conf, on Sof. Maintenance and Reengineering. CSMR 01, pages39-46, Washington, DC, USA, 2001. IEEE Computer Society.
[29] R.K. Bandi, V.K. Vaishnavi and D.E. Tuk. Predicting maintenance performance using object- oriented design complexity *met*rics. IEEETSE, 29: 77-87, 2003.
[30] L.M. Hakik, R. El Harti. Comparison of Results Obtained by Application of Techniques Based on Formal Concept Analysis and Oriented Graph for a Remodulaisation Software Architecture Composed of Classes and Packages. IJCSNS International Journal of Computer Science and Network Security .Vol. 14 No. 2 . Journal ISSN : 1738-7906. January 2014.

Lala Madiha Hakik received the Maitrise in Computer Engineering, from Hassan 1st University, FST, Settat, Morocco in 2005, Specialized Master in Software Engineering, Montpellier -2- University, France in 2009, She is a PHD Student in Computer Science specialized in Software Engineering, University Hassan 1st , FST, Settat, Morocco, 2014. Members of the specific scientific committee and editorial review on World Academy of Science, Engineering and Technology. International association for the engineers and computer scientists member (IAENG member).

Rachid El Harti received the PHD in Mathematics and applications from Mohammed V University, Morocco in 1993, Full professor, Hassan 1st University, Morocco