A Probabilistic Approach for Change Impact Prediction in Object-Oriented Systems

M.K. Abdi*, H. Lounis**, H. Sahraoui*

** Département d'Informatique et de Recherche Opérationnelle,*
Université de Montréal,
CP 6128 succ Centre-Ville, Montréal QC H3C 3J7, Canada
{abdimust, sahraouh}@iro.umontreal.ca
*** Département d'Informatique, Université du Québec à Montréal*
Case postale 8888, succursale Centre-ville, Montréal QC H3C 3P, Canada
lounis.hakim@uqam.ca

Abstract Several non probabilistic approaches were proposed in the literature to analyze and predict change impact in Object-Oriented (OO) systems. Different aspects were considered in these studies and several experiments were conducted to check some hypotheses. However, causality relation between software internal attributes and change impact still misses convincing explanations. In this paper, we propose a probabilistic approach using Bayesian networks to answer to this problematic of change impact analysis and prediction in OO systems. The built probabilistic model is tested on data extracted from a real system. The running of different scenarios on the network, globally confirm results already found in previous studies.

1 Introduction

Systems modification is a difficult task that has an impact on systems becoming [24]. Change effects must be considered. A small change can have considerable and unexpected effects on the system. Risks incurred during a modification are related to the consequence of a given change impact. When modularity is adequately used, it limits the effects relating to changes. Nevertheless, change impacts are subtle and difficult to discover; designers and maintainers need mechanisms to analyze changes and to know how they are propagated in the whole system.

The main motivation of our work is to improve the maintenance of object-oriented systems, and to intervene more specifically on change impact analysis. By identifying the potential impact of a modification, one reduces the risk to deal with expensive and unpredictable changes. Consequently, we try to give more explanations on real and responsible factors for change impact and its evolution. Among several models of representation, Bayesian networks (BNs) constitutes a particular quantitative approach which can integrate uncertainty within reasoning [20] offering thus explanations that are close to reality. Moreover, with BNs, it is also possible to exploit experts' judgements to anticipate predictions, in our case, on change impact. In addition, BNs have the capacity of incremental training on data. This is true as well for parameters training as for structure training, facilitating the model evolution. This characteristic will contribute to the improvement of Bayesian network structure and parameters, by the acquisition of new data.

In this paper, section 2 presents various works related to change impact analysis. Our approach is presented in the third section. We start by presenting the principal stages of our approach, followed by a short recall on BNs. Then we illustrate gradually how to build the graph (BN) within the framework of our experimentation. After that, we explain the parameters assignment (probabilities) to network nodes. Section 4 concerns network execution and results discussion. Finally, our work perspectives are discussed in the conclusion.

2 Related works

Several studies were conducted on change impact. Thus, Han [12] developed an approach for computing change impact on design and implementation documents. This approach considers the original representation of software artefacts (classes) rather than a model of extracted system separately. The artefacts dependencies imply inheritance, aggregation and association. Furthermore, impacts are not defined in a formal way. On another side, Lindvall [19] identified the most common and frequent changes in C++, so that the change models can be specified to help developers to envisage the future needs. In [4], Antoniol and al. predicted evolving object-oriented systems size starting from the analysis of the classes impacted by a change request. They predicted changes size in terms of added/modified lines of code. Kung and al [16], interested by regression testing, developed a change impact model based on three links: inheritance, association, and, aggregation. They also defined formal algorithms to calculate all the impacted classes including ripple effects. Lee and Offutt examined in [17] and [18] the effects of encapsulation, inheritance, and polymorphism on change impact; they also proposed algorithms for calculating the complete impact of changes made in a given class. However, some changes, implying for instance inheritance and aggregation, were not completely covered by their algorithms.

In [7], impact analysis was made to reduce the costs and duration of regression tests. The study was made starting from a dependence graph. Briand and al. in [5], tried to see if coupling measures, capturing all kinds of collaboration between classes, can help to analyze change impact. This study, (i) showed that some coupling metrics, related to aggregation and invocation, are connected to ripple effect, and, (ii), it allows performing dependence analysis and reducing impact analysis effort. In [6], [14] and [15], a change impact model was defined at an abstract level, to study the changeability of object-oriented systems. The adopted approach uses characteristic properties of OO systems design (complexity, cohesion, coupling, etc.), measured by metrics, to predict changeability. According to a different perspective, Sahraoui and al. studied in [22] the impact of refactoring on structure and thus on structural metrics. This study made it possible to determine the refactorings that can improve or deteriorate certain structural properties. Recently, in [1], [2], and [3], the authors also showed that coupling, measured by some metrics, influences change impact.

On the other hand, in [9], Fenton and Neil show well the advantages of the causal-modelling approach using Bayesian networks compared to the naive regression-based approach. In other works [10], [11], and [21], they also prove through case studies that Bayesian nets can provide relevant predictions, as well as incorporating the inevitable uncertainty, reliance on expert judgement, and incomplete information that are pervasive in software engineering. In this work, we try to explore this way of research and thus show the advantages of probabilistic approach using Bayesian nets compared to the approach adopted in our former work [1], [2], and [3].

In the following section, we present our approach (proposition) by explaining its different stages.

3 Proposition

The main stages of our approach are the following:

1- Graph structure construction (BN) starting from practical knowledge (empirical studies)
2- Parameters affectation (node probability table, fuzzy logic).
3- Bayesian inference (algorithms, tools)
4- Results

The first two stages are explained in the present section while the two last stages are presented in section 4. In order to facilitate the comprehension of used concepts in our approach, a recall on the basic concepts of Bayesian networks is essential.

3.1 Recall on Bayesian networks

BNs are based on the Bayes theorem. This theorem describes the relations which exist between simple and conditional probabilities. If A and B are two events and if we know the probability of A, of B and B knowing A, the Bayes theorem allows to determine the probability of A knowing B:

$$P(A/B) = \frac{P(B/A)P(A)}{P(B)}$$

BNs are the result of a merging between graph theory and probability theory [20]. A BN is a causal graph where:

- Nodes represent random variables. A random variable has some states, for example “Yes” and “No”, and a distribution probability for these states, where the sum of probabilities of all states must be equal to 1. Thus, a BN model is in conformity with the standard axioms of probability theory.

- Oriented edges define causal relations between nodes. An edge goes from a parent node towards a child node. Parent nodes which affect the same child node must be independent variables. Each node is related to a Node Probability Table (NPT), which models uncertain relation between the node and its parents. Tables of conditional probability related to BN nodes determine the force of the graph bonds and are used to calculate the distribution probability of each node in BN. This is carried out by specifying the conditional probability of a node knowing all its parents: p(X | A, B), X being the child node of A and B. If a node has no parent, a probability table would be associated for this node. Usually, NPTs are generally created by using a mixture of empirical data with experts judgement. In this causal graph, the cause and effect relationships between the variables are not deterministic, but probabilistic. Thus, observation of a cause or several causes doesn't involve systematically the effect or effects which depend on them, but modifies only the probability of observing them. The particular interest of BNs is to hold account as well of experts knowledge (in the graph or its structure) as of experiments contained in data (parameters).

3.2 Graph construction (BN)

Generally, the BN construction is done in two stages: produce the suitable graph because this model is sensitive to the type of applied reasoning, then affect probability values to network nodes [20]. The affectation of these values is done according to domain experts or starting from empirical studies. At this level, it is important to check that the parents nodes which affect the same child node are independent variables. Moreover, in order to respect the BNs construction formalism, during the introduction of bonds between nodes, it is necessary to check the absence of cycles between network nodes.

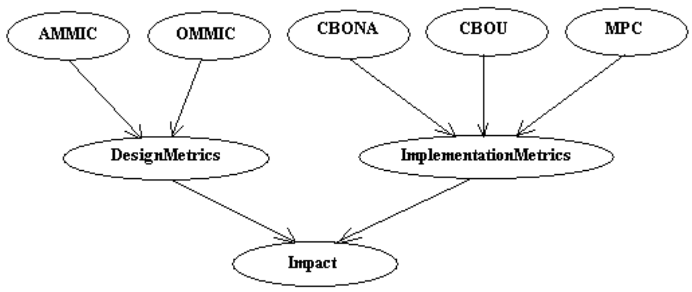

Figure 1. Change impact network

As already stated (in section 2), we checked in [1], [2], and [3], the hypothesis claiming that coupling influences change impact in an object-oriented systems. However, if we consider at the same time all metrics measuring the various facets of coupling between classes, the BN construction is likely to be hard and its structure complex. In addition, the results affirm that among the ten selected metric (see table 1), measuring this architectural property, five metrics are effectively relevant to change impact. Some of these metrics are regarded as design metrics (*AMMIC* and *OMMIC*), others are considered as implementation metrics (*MPC*, *CBOU*, and *CBONA*). The figure 1 above presents the graph expressing this knowledge in the form of a BN. Let us note that in a BN, the relation between parents and child nodes are causal (case of *Impact* node) or definitional (case of *DesignMetrics* node).

Metrics	Definition
RFC	Response For a Class: number of methods called upon in response to a message.
MPC	Message Passing Coupling: number of messages sent by a class in direction of the other classes of the system.
CBOU	CBO Using: refers to the classes used by the target class.
CBOIUB	CBO Is Used By: refers to the classes using the target class.
CBO	Coupling Between Object: number of classes with which a class is coupled.
CBONA	CBO No Ancestors: CBO without considering the classes ancestors.
AMMIC	Ancestors Method–Method Import Coupling: number of parents classes with which a class has an interaction of the method-method type and a coupling of the type IC.
OMMIC	Others Method–Method Import Coupling: number of classes (others that super classes and subclasses) with which a class has an interaction of the method-method type and a coupling of the type IC.
DMMEC	Descendants Method–Method Export Coupling: number of subclasses with which a class has an interaction of the method-method type and a coupling of the type EC.
OMMEC	Others Method–Method Export Coupling: number of classes (others that super classes and subclasses) with which a class has an interaction of the method-method type and a coupling of the type EC.

Table 1. The selected coupling metrics

3.3 Parameters affectation

To affect probabilities to the nodes, it is necessary to distinguish two types of variable in BN: entry variables and intermediate variables. The entry nodes probabilities are directly deduced from measurements of these variables starting from a given test system. In our case, we chose a program analysis toolbox system, called BOAP, and, developed at the computer science research center of

Montreal (CRIM) [8]. It is a set of integrated software tools, which allow an expert to evaluate some software qualities, e.g., conceptual or structural weaknesses, too complex instructions, etc. We considered the BOAP system in its version 1.1.0; it is written in Java and contains 394 classes. The metric considered in this work are extracted from this system.

Entry nodes. In our network (figure 1), the entry nodes represent the different metrics. All these entry variables are quantitative variables which have measurable numerical values. The number of possible values for these variables can be infinite. That depends of course on the considered test system. In order to facilitate the probabilities definition, these variables are initially transformed into discrete variables having a limited number of values. This transformation can be accomplished by application of fuzzy logic. Indeed, the fuzzy partitioning process replaces the various values of a metric by a set of functions which represent the membership degree (or adhesion) of each value to the various fuzzy labels (often "small", "average" and "large"). The fuzzy partitioning generalizes the regrouping methods by groups allowing a value to be partially classified in one or more groups at the same time. The adhesion or the value membership is distributed in all groups. However, empirically, we can determine the optimal number of groups with statistics known under the name of Dunn partition coefficient F_k. This coefficient indicates us how to gather with a better way a data set in various groups [23]. The more the Dunn coefficient is high, the more the fuzzy subsets coincide classical logic sets. Therefore, the optimal number of groups is that which maximizes F_k. The Dunn partition coefficient is calculated according to the formula:

$$F_k = \frac{1}{N} \sum_{i=1}^{N} \sum_{g=1}^{k} u_{ig}^2$$

N being the full number of observations (data), g the index for a group, k the number of groups and u_{ig} the value or the membership degree of a given object to a group.

Table 2 presents the results of fuzzy partitioning with 2 and 3 groups for the *AMMIC* metric. These results show that with two groups the Dunn coefficient is 0.8171413 and with three groups it is equal to 0.7768965. Therefore, for this metric, the partitioning in two groups is retained. Moreover, it is the same number of groups which was retained following the fuzzy partitioning tests for the four others metric. We used for that the statistics software S-plus (version 8.0) [13].

Table 3 gives an example of NPT for *AMMIC* node. It is about an example of value measured (equal to 25) for the *AMMIC* metric. To this value correspond two membership degrees (0.4349570 and 0.56504302) in the two fuzzy subsets. These membership degrees constitute the probabilities which are used to define the NPT of *AMMIC* node.

```
                       *** Fuzzy Partitioning ***

 Membership coefficients:      Membership coefficients:
 numeric matrix: 394 rows, 2   numeric matrix: 394 rows, 3 columns.
 columns.                               [,1]        [,2]        [,3]
         [,1]       [,2]
  1 0.9873814 0.01261863        1 0.99358023 0.004435426 0.001984347
  2 0.9873814 0.01261863        2 0.99358023 0.004435426 0.001984347
  3 0.9873814 0.01261863        3 0.99358023 0.004435426 0.001984347
  …       …          …          …       …          …           …

 392 0.9873814 0.01261863      392 0.9935802 0.004435427 0.001984347
 393 0.9873814 0.01261863      393 0.9935802 0.004435427 0.001984347
 394 0.9873814 0.01261863      394 0.9935802 0.004435427 0.001984347

 Coefficients:                 Coefficients:
  dunn_coeff normalized         dunn_coeff normalized
   0.8171413  0.6342827          0.7768965  0.6653447
```

Table 2. Example of fuzzy partitioning for *AMMIC*

Small	0.43
Large	0.57

Table 3. The NPT of *AMMIC* entry node

Intermediate nodes. The intermediate nodes are not directly measurable. They are defined or influenced by their parent nodes. For each intermediate node Cc which has possible values $\{V_{c1}, \ldots V_{ck}, \ldots V_{cn}\}$ and has parents $\{C_{p1}, \ldots C_{pi}, \ldots C_{pm}\}$ with possible values $\{V_{ci1}, \ldots V_{cij}, \ldots V_{cil}\}$, we need to define a table which gives the probabilities for all possible combinations of values:

$$P\ (V_{ck} \mid V_{p1j},..., V_{pmj})$$

These probability values can be adjusted by using machine learning starting from the sample data or the treated cases. A parent can influence positively or negatively his child nodes. The probability distributions are affected according to the importance or the weight of each parent for the child node. At the beginning, to derive NPT it is necessary to consider the weight of each parent node in definition or influence of its child node. For that, NPTs are initially given starting from studies in the field and experts opinions. For instance, the *DesignMetrics* variable is defined by its two parents *AMMIC* and *OMMIC*. It is a question of finding the conditional probability of *DesignMetrics* node: p (*DesignMetrics*| *AMMIC*, *OMMIC*). However, like the relation between the parent nodes *AMMIC* and *OMMIC* and their child node *DesignMetrics* is definitional, the strong presence of these metrics also defines the strong presence of *DesignMetrics*. A possible scenario for the *DesignMetrics* node NPT is presented in table 4:

AMMIC	Small		Large	
OMMIC	Small	Large	Small	Large
Oui	0.2	0.4	0.4	0.8
Non	0.8	0.6	0.6	0.2

Table 4. The *DesignMetrics* intermediate node NPT

A reasoning which can be applied is the following: if the number of classes (others that super-classes and subclasses) with which this class has an importation interaction of the method-method type is small (*AMMIC* small), and the number of parents classes with which this class has an importation interaction of the method-method type is small also (*OMMIC* small), the design metrics presence probability in such a system is weak or small. Therefore, the probability of the state “Yes” in the probability table of *DesignMetrics* node can be 20%. Conversely, if *AMMIC* is large, and *OMMIC* is large also, the probability of the state “Yes” of *DesignMetrics* node can be 80%. It is important to recall here that there are obviously other metrics (other than those considered in this study) and which are defined like design metrics or implementation metrics, and consequently, can positively or negatively influence change impact.

4 Bayesian Network execution

Once the graph structure and all NPTs are defined, we can proceed with the Bayesian inference. It results an update of conditional probabilities of all nodes. We have used the BNJ (Bayesian Network tools in Java) environment to achieve this goal. BNJ is a set of open source software tools intended for research and development by using graphic probabilities models. It is written in Java and is available on the web[1].

Let us recall that our experimentation was made on the BOAP test system (version 1.1.0) which contains 394 classes, or 394 instances. For the network execution, we will randomly choose an instance from which we take the metric values corresponding to entry nodes. As soon as the probabilities distributions are updated for introduced values, we will have an estimate in the form of probability for the various states assignated to the *Impact* node (figure 2).

Having affected three states «Weak», «Average», and «Strong» to the *Impact* node, and with the used input data (see figure above), we can conclude that the change impact has 43% of probability of being “Strong”. The possibility of processing scenarios of the form « what will occur if… ?», that Bayesian networks offer, allows to identify potential problems and actions to be undertaken for improvement.

1. http://bnj.sourceforge.net/

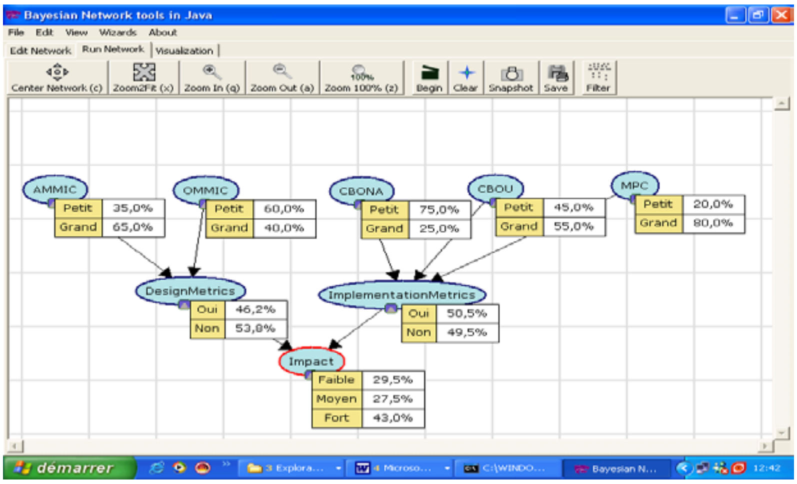

Figure 2. Change impact network after scenario 1

The scenario 2 execution shows that by decreasing the metrics values *CBONA* and *CBOU*, change impact weakens more (its probability of being "Weak" grows from 29,5% to 34,8%). Conversely, the scenario 3 execution shows that by increasing the *CBONA* and *CBOU* metrics values; the change impact becomes increasingly strong. The probability of the «Strong» state moves from 37,8% to 47,4%. Figure 3 illustrates this result.

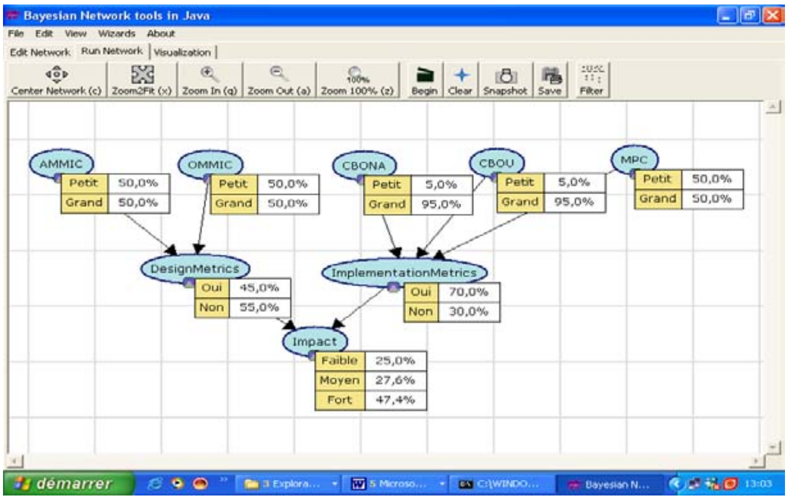

Figure 3. Change impact network after scenario 3

Finally, the last scenario execution shows that by maintaining the values of *CBONA* and *CBOU* metrics and by increasing the *AMMIC* one, change impact becomes little stronger. The probability of the «Strong» state grows from 37,8% to 41,5%.

Discussion

Results obtained in the second and third scenarios confirm those already found in our former work [1], [2], and [3], by using a non probabilistic approach (see respectively rule 1 and rule 2 of figure 4). For example, the scenario 3 result expressing that *CBONA* and *CBOU* metrics influence positively change impact, corresponds to the result illustrated by the causality rule 2 [2]:

Rule 1 : CBONA $\leq$ 3.5
CBOU $\leq$ 0.5
$\rightarrow$ impact: Weak (0.46)

Rule 2 : CBONA > 3.5
CBOU > 36.5
$\rightarrow$ impact: Strong (0.48)

Rule 3 : CBONA $\leq$ 3.5
CBOU $\in$]0.5,1.5]
AMMIC $\leq$ 0.5
$\rightarrow$ impact: Weak (0.54)

Rule 4 : CBONA $\leq$ 3.5
CBOU $\in$]0.5,1.5]
AMMIC > 0.5
$\rightarrow$ impact: Weak (0.76)

Figure 4. Causality rules examples

On the other hand, the scenario 4 result does not confirm one of our results (see rules 3 and 4 of figure 4) found before in [1] and [2]. Indeed, by maintaining the *CBONA* and *CBOU* metrics values small, and by increasing the *AMMIC* value, change impact does not become more weak. Its probability of being "Weak" was 34,8% then it was reduced to 30,8% whereas in theory, it must increase. In our opinion, that could be explained by the fact that change impact can be positively or negatively influenced by other metrics, other than those considered in the present study, or also, by other factors, like system size, complexity, etc.

5 Conclusion

We proposed in this article a probabilistic approach using Bayesian networks to analyze and predict change impact in object-oriented systems. A thorough study and a general synthesis of various former works dealing with this subject were initially essential. To verify our approach, we took again a correlation hypothesis between coupling and change impact already verified in former works. The experimentation was made on BOAP system. It contains 394 classes. The results of our empirical studies ([1], [2] and [3]) were useful for the graph structure construction (Bayesian Network). Thereafter, we defined the NPTs of entry and intermediate nodes. We used fuzzy logic to derive probabilities values starting from a set of measures (variables values or entry nodes).

The network execution and the creation of several scenarios enabled us to make predictions on change impact. The results of the second and third scenario confirmed results already found with other non probabilistic approach. On the other hand, the results of the fourth scenario contradict one of our results found before [2]. That leads us to search a hypothesis explaining this last result.

Finally, we are in the process of considering further experiments on other systems by including other coupling measurements, other architectural properties, or other factors which could supplement or better explain this causality relation.

References

[1] M.K Abdi, H. Lounis, H. Sahraoui: "Using Coupling Metrics for Change Impact Analysis in Object-Oriented Systems" In QAOOSE 2006 Proceedings, 10th ECOOP Workshop on Quantitative Approaches in Object-Oriented Software Engineering, 3 July 2006 - Nantes, France

[2] M.K Abdi, H. Lounis, H. Sahraoui: "Analyzing Change Impact in Object-Oriented Systems " In proceedings of the 32nd EUROMICRO Software Engineering and Advanced Applications Conference, Cavtat/Dubrovnik (Croatia), August 29-September 1, 2006.

[3] M.K Abdi, H. Lounis, H. Sahraoui, M.K Rahmouni : "Vers une approche d'analyse de l'impact du changement dans un système à objets", dans revue "L'Objet", volume 13 – N° 1/2007, Pages 147-169, Éditions Hermès.

[4] G. Antoniol, G.Canfora, A. D. Lucia, "Estimating the size of changes for evolving Object-Oriented Systems : a Case Study" *in Proceedings of the 6th International Software Metrics Symposium*, pages 250-258, Boca Raton, Florida, Nov 1999

[5] L.C. Briand, J. Wüst, H. Lounis, "Using Coupling Measurement for Impact Analysis in Object-Oriented Systems" *in proceedings of the International Conference on Software Maintenance ICSM'99,* Oxford, England, August 30 – September 3, 1999.

[6] M.A. Chaumun, H. Kabaili, R.K. Keller and F. Lustman. *"A Change Impact Model for Changeability Assessment in Object-Oriented Software Systems".* In Proceedings of the Third Euromicro Working Conference on Software Maintenance and Reengineering CSMR'99, pages 130-138, Amsterdam, The Netherlands, March 1999.

[7] R. Cantave, Abstractions via un modèle générique d'application orientée objet, Master's thesis, Université Laval, Canada, Avril 2001

[8] E. Alikacem, H. Snoussi, *"BOAP 1.1.0 : Manuel d'utilisation",* CRIM, Janvier 2002.

[9] N.E. Fenton and M. Neil, "Software Metrics: Roadmap", in 'The Future of Software Engineering' (Editor: Anthony Finkelstein) 22nd International Conference on Software Engineering, ACM Press ISBN 1-58113-253-0, pp.357-370, 2000.

[10] N.E. Fenton and M. Neil, "The Jury Observation Fallacy and the use of Bayesian Networks to present Probabilistic Legal Arguments", Mathematics Today (Bulletin of the IMA, 36(6)), 180-187, 2000.

[11] N.E. Fenton and M. Neil, "Making Decisions: Using Bayesian Nets and MCDA", Knowledge-Based Systems 14, 307-325, 2001.

[12] J. Han, "Supporting Impact Analysis and Change Propagation in Software Engineering Environments" *in Proceedings of the STEP'97*, London, England, pages 172-182, July 1997.

[13] Insightful Corporation, Seattle, WA, *S-PLUS® 8 for Windows® User's Guide*, Copyright © 1987-2007.

[14] H. Kabaili, R.K. Keller, F. Lustman, and G. Saint-Denis. Class Cohesion Revisited: An Empirical Study on Industrial Systems. In *Proceedings of the Workshop on Quantitative Approaches in Object-Oriented Software Engineering*, pages 29-38, Cannes, France, June 2000

[15] H. Kabaili, *"Changeabilité des logiciels orientés objet : propriétés architecturales et indicateurs de qualité"*, PhD thesis, Université de Montréal, Canada, Janvier, 2002

[16] D.C. Kung, J. Gao, P. Hsia, J. Lin, Y. Toyoshima, "Class firewall, test order, and regression testing of object-oriented programs" *in Journal of Object-Oriented Programming*, Vol. 8, No. 2, pages 51-65, May 1995.

[17] M. Lee, A.J. Offutt, "Algorithmic Analysis of the Impact of Changes to Object-Oriented Software" *in Proceedings of the ICSM'96*, pages 171-184, 1996.

[18] M. Lee, Change Impact Analysis for Object-Oriented Software, PhD thesis, George Mason University, Virginia, USA, 1998

[19] M. Lindvall, "Measurment of change : Stable and Change-Prone Constructs in a commercial C++ System" *in Proceedings of the 6th International Software Metrics Symposium,* pages 40-49, Boca Raton, Florida, Nov 1999

[20] P. Naïm, P. Wuillemin, P. Leray, O. Pourret, A. Becker, "Réseaux bayésiens", Edition Eyrolles, 2004

[21] Neil M, Fenton NE, Nielsen L, "Building large-scale Bayesian Networks", The Knowledge Engineering Review, 15(3), 257-284, 2000.

[22] H.A. Sahraoui, R. Godin, T. Miceli, "Can metrics help to bridge the gap between the improvement of OO design quality and its automation ?", *in Proceedings of the International Conference on Software Maintenance (ICSM'00)*, 2000

[23] E. Trauwaert, On the meaning of Dunn's partition coefficient for fuzzy clusters, *Fuzzy Sets and Systems,* Vol.25, No 2, pp 217-242, 1988.

[24] N. Wilde, R. Huitt, "Maintenance support for object-oriented programs" i*n IEEE Transactions on Software Engineering,* Vol. 18, Issue 12, Pages 1038–1044, Dec 1992.