Metrics for Active Database Maintainability*

Oscar Díaz[1] and Mario Piattini[2]

[1] Departamento de Lenguajes y Sistemas Informáticos,
University of the Basque Country - San Sebastián, Spain
jipdigao@si.ehu.es
[2] Departamento de Informática,
University of Castilla-La Mancha - Ciudad Real, Spain
mpiattini@inf-cr.uclm.es

Abstract. Databases are becoming more complex and it is necessary to measure schemata complexity in order understand, monitor, control, predict and improve software development and maintenance projects. Active databases are a case in point where several reports warned the difficulties to cope with large rule sets. This paper proposes three different metrics for measuring active databases complexity, based on the difficulty to ascertain the causes that make a given rule to be triggered. The measurement theory is used to characterise these metrics.

1 Introduction

Software engineers have been proposing large quantities of metrics for software products, processes and resources. Most of these metrics focus on programme characteristics neglecting databases. This disregard could be explained as databases have been until recently just “simple files/tables” with minor contribution to the complexity of the overall system. With the new database generation however, this is no longer the case. New applications have fuelled the appearance of the “third database generation”, where new data types, rules, generalisations, complex objects and functions are being supported within the database realm.

The focus of this paper are active database management systems (DBMS), that is, DBMS that allow the definition and management of event-condition-action rules. This mechanism enables the DBMS to respond automatically to events that are taking place either within or outside the database itself. This mechanism is currently available in a large number of commercial products (Oracle, Sybase, DB2, SQL-Server, Interbase), and rules are reckoned to be useful for a large number of applications. Despite its usefulness, early reports show that a few hundred rules represent a real maintenance problem. Similar disquiet was pointed out at the closing panel at the RIDE-ADS workshop dedicated to Active Database Systems [1]. As these reports suggest, the lack of maintainability is a

* This work is partially supported by the MANTICA Project (CICYT/European Union 1FD97-0168).

main obstacle for active system to become widely used. Maintainability is contributed by three factors: understandability, modifiability and testability, which in turn are influenced by complexity [2].

This paper addresses the complexity of large rule sets by means of three measures: the triggering potential, the number of anchors, and the distance of a rule. These measures are characterised using the measurement theory, particularly the formal framework proposed by Zuse [3]. The rest of the paper is organised as follows. Section 2 briefly introduces those aspects of active DBMS significant for this work. Section 3 presents the different complexity measures proposed. We characterise the proposed metrics in Section 4.

2 Characterization of Active Databases

Active DBMS are able to react to significant events. These systems provide a way to describe events and their associated reactions (i.e. the knowledge model) as well as the runtime strategy to process and combine this active behaviour (i.e. the execution model).

A common approach for the knowledge model uses event-condition-action rules (ECA rules) that have an event which describes a happening to which the rule may be able to respond, a condition that examines the context in which the event has taken place, and an action which describes the task to be carried out by the rule if the relevant event has taken place and the condition has evaluated to true. The event part can be a primitive event (e.g. a database operation) or a composite event, i.e., a combination of primitive or composite events. For the purpose of this paper, the cardinality of the rule's event correspond to the number of primitive events that are (either direct or indirectly) referred to in the event part of the rule, regardless of the composite operator used.

The execution model specifies how a set of rules is treated at run-time. Among other aspects, this model includes the coupling mode that determines when the condition (action) is evaluated (executed) relative to the triggering (evaluation) of the event (condition). In this paper, only the immediate coupling mode is considered where the condition (action) is evaluated (executed) immediately after the event (condition). The question of what happens when events occur during the execution of the rule's action is addressed by the cycle policy. This paper, considers a recursive model where events risen during action execution are immediately taken into account. This causes the current rule's action to be suspended, so that any rules monitoring these events can be processed at the earliest opportunity.

Early reports on the use of active systems point out the feeling of lack of control experimented by users. Indeed, the audience at the RIDE-ADS'94 closing panel qualified as infeasible ECA-rule applications which have *more than 7 rules or more than 5 layers of rule triggering* [1]. It is worth noticing that the obstacle identified during this workshop was not the unavailability of appropriate mechanisms, but the complexity of the already available systems.

3 Measures for Active Datatabase Complexity

Design product metrics can be sub-divided into intra-module and inter-module metrics. Likewise, rule complexity can be characterised as intra-rule complexity where the rule in isolation is measure, and inter-rule complexity where the implicit interaction among rules is measure. Experience in building active systems, suggests that it is the degree of interactions rather than the number or complexity of the rules themselves, what affects this sense of lack of control felt by the user. More concretely, two aspects are felt to be specially significant in the interaction of rules: the width and depth of the argumentation. The former reflects the intuition that the larger is the flow of events that conforms the rules circumstance, the more difficult will be to understand these rules. Of course, rules being fired by composite events are potentially more intricate that those with a single event. However, what our experience shows out is that a factor that complicates things even more is when the very same event type that participates in the rule's event is produced in distinct places. This means that the rule's circumstance can raise in different contexts. As a result, the user has a tougher job at ascertaining which of the potential circumstance makes the rule to be triggered. As for the depth of the argumentation, it refers to the intricacy of the line of reasoning that connects the rule with the context where its circumstance has been produced. The depth and number of threads required to encompass the rule's circumstance certainly affects its complexity.

To formalise these intuitions, the notion of a triggering graph is used [4]. A triggering graph is a pair $< S, L >$ where S represent the set of ECA rules, and L is a set of directed arcs where an arc is drawn from Si to Sj if Si's action causes the happening of an event occurrence that participates in Sj's events. This notion of triggering graph is slighted modified for our purposes in two aspects. First, arcs are weighted by the number of potential event occurrences produced by the triggering rule (i.e. Si) that could affect the rule triggered (i.e. Sj's event). Second, the nodes S are extended with the set of transactions T. A transaction is a set of (database) actions where any of these actions could correspond to an event triggering one or more rules. Therefore, T nodes will have outgoing links but never incoming links as we assume that a transaction can never be fired from within a rule's action or another transaction.

The intuitive notions identified previously can now find correspondence as triggering graph measures, namely:

- **NA**, the minimum number of anchors required to encompass the whole set of potential causes of Si. An anchor is a transaction node of the triggering graph which has a link (either direct or transitively) with at least one cause of Si. The intuition is that each of these anchors represents a line of reasoning to be followed to understand the rule's circumstance.
- **D**, the distance which accounts for the length of the longest path that connects Si with any of its anchors. The intuition is that this measure reflects the intricacy of the line of reasoning as the number of inferences which are required to connect the ultimate cause with its effect (i.e. the triggering of Si).

– **TP**, the triggering potential. Given a triggering graph <S,L>, and a node of the graph, rule Si, the number of causes of Si, is the sum of weights of the incoming arcs arriving to Si. The triggering potential for a rule R is the quotient between the number of potential causes of Si, and Si's event cardinality. This measure attempts to reflect the width of Si's circumstance by giving an indication of the potential different circumstances which can make Si to be triggered.

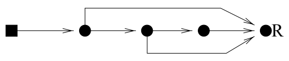
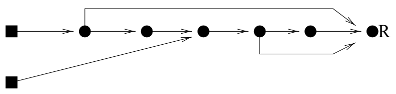
	Trig. pot.	Nª Anc.	Dist.
	3	1	4
	3	2	5

Fig. 1. Measuring different triggering graphs.

Figure 1 illustrates the previous measures for distinct triggering graphs where the event cardinality for rule S is one.

4 Characterization of Active Database Complexity Metrics

The framework described in [3] is used to describe the properties of the metrics defined above. This framework is based on an extension of the classical measurement theory, which gives a sound basis for software measures, their validation and criteria for measurement scales.

For ECA rules, the Empirical Relational System, $A = (S, \bullet >=, o)$ could be defined as follows:

1. S is a non-empty set of rules
2. $\bullet >=$ is the empirical relation "more or equal complex than" on S
3. o is a closed binary (concatenation) operation on S such that concatenation of rules S1(E1, C1, A1) and S2(E2, C2, A2) produces a rule S3, where:
   – event E3 = E1 OR E2
   – condition C3 = (C1 AND e1) OR (C2 AND e2), where ei is a Boolean variable where true (false) indicates the occurrence (absence) of the Ei event
   – action A3 can be defined as: IF e1 THEN A1, IF e2 THEN A2.

Axioms	Anchor Number	Distance	Trig. Potential
Axiom 1	yes	yes	yes
Axiom 2	yes	yes	no
Axiom 3	yes	yes	yes
Axiom 4	yes	yes	yes
Axiom 5	no	yes	no
Axiom 6	no	no	no
Ind. Cond. 1	no	yes	no
Ind. Cond. 2	no	no	no
Ind. Cond. 3	no	yes	no
Ind. Cond. 4	no	no	no
MRB 1	yes	yes	yes
MRB 2	yes	yes	yes
MRB 3	yes	yes	no
MRB 4	no	no	no
M4B 5	yes	yes	yes

Table 1. Characterization of rule's circumstance measures.

The outcome is that the proposed measures do not represent `an extensive structure` but they can be characterised above the ordinal scale by fulfilling some of the properties of modified relations of belief. Table 4 summarises the properties fulfilled by the three metrics following the Zuse framework.

References

[1] J. Widom. Research issues in active database systems: Report from closing panel at ride-ads' 94. In *SIGMOD RECORD*, volume 23 (3), pages 41–43, 1994.

[2] H.F. Li and W.K. Chen. An empirical study of software metrics. *IEEE Trans. on Software Engineering*, 13(6):679–708, 1987.

[3] H. Zuse. *A Framework of Software Measurement*. Walter de Gruyter (Berlin), 1998.

[4] A. Aiken, J.M. Hellerstein, and J. Widom. Static analysis techniqies for predicting the behaviour of active database rules. *ACM Transactions on Databases*, 20(1):3–41, 1995.