Ontology-based Correlation Engines

Ljiljana Stojanovic[1], Andreas Abecker[1], Nenad Stojanovic[2], Rudi Studer[1,2]
[1]FZI - Research Center for Information Technologies at the University of Karlsruhe
[2]Institute AIFB, University of Karlsruhe
Stojanovic@fzi.de; Abecker@fzi.de; nst@aifb.uni-karslruhe.de; studer@aifb.uni-karslruhe.de

Abstract

Correlation engines are autonomic computing systems that perform the automated, continuous analysis of enterprise-wide event data based on user-defined, configurable rules in order to detect threats and protect a system from them. In this paper we discuss the run-time advantages of using ontologies as a conceptual backbone for describing knowledge processed by correlation engines.

1. Introduction

Correlation engines are autonomic computing systems that perform the automated, continuous analysis of enterprise-wide, normalized and real-time event data based on user-defined, configurable rules. They are realized according to the **MAPE** (**M**onitor **A**nalyze **P**lan **E**xecute) model [1], which abstracts management architecture into four common functions: (1) collect the data, (2) analyze the data, (3) create a plan of action, and (4) execute the plan. The MAPE model assumes the existence of a common knowledge element that represents the knowledge about a domain that is shared between four components of the MAPE model. However, the MAPE model does not provide guidelines how to represent the knowledge about a domain, how to acquire data about problems in a domain, and how to use that information to resolve problems without putting humans in every possibility. Consequently, each correlation engine has its own knowledge model.

In our previous work [2] we proposed a general knowledge model for correlation engines by abstracting/describing knowledge hidden in the knowledge element on the conceptual level. We showed that this data abstraction could be realized by extending and combining correlation engines with ontologies. Although we used the "eAutomation" correlation engine [3] as an example, the similar strategies can be applied for other engines as well.

The approach can be summarized as follows: (i) the model of the "eAutomation" engine is transformed into the eAutomation ontology; (ii) „hidden" (hard-coded) knowledge embedded in the "eAutomation" engine is translated into a set of rules and is used in typical inferencing tasks.

This explicit representation of the semantics of data enables correlation engines to provide a qualitatively new level of services. In this paper we discuss these benefits.

2. Run-time benefits

The resolution of a request in an ontology-based system is realized as an inference process, which is "recorded" as a derivation tree. It can be interpreted at the level of an ontology-based correlation engine, where a request corresponds to a problem that has to be resolved and a solution corresponds to an action that has to be triggered. By applying the derivation tree analysis in the context of the correlation engines, the following benefits can be achieved:

1. *Justification* means the generation of the human-understandable descriptions of the inference process. The explanation about the reasons for suggesting a certain corrective action can be presented to the administrators in order to inform them or to a software agent that is responsible for recovering from mistakes. In that way, the confidence of the administrators in the correlation engine can be significantly improved and the software agent can optimize its actions.

2. *Ranking* means to determine the relevance of results in the case that lots of results are retrieved. The estimation of the relevance for a result can be obtained by analyzing the complexity of the derivation tree for that result. A broader tree is an indicator for a lot of support for that result; a deeper tree indicates less confidence in that result.

This strategy can be applied in the assessment of the importance of the proposed action in the autonomic computing systems. An action that can be obtained as a result of the more independent rules is more important and therefore has to be applied firstly. Considering the tree depth, an action that is recommended directly (i.e. through only one rule) is more important that an action that is suggested indirectly (i.e. through the composition of the several rules), since the second case requires more conditions to be fulfilled.

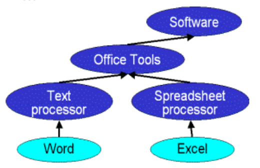
3. *Gap analysis* is related to the discovery of problems in the domain knowledge in case that no result is retrieved. In the autonomic computing systems, in order to determine which events to monitor and how to analyze them, administrators must be familiar with the operational parameters of each managed resource and of its events. Furthermore, there is no administrator who possesses all knowledge about a domain. Therefore, the development of a model for a concrete domain is a bottleneck. The analysis of the broken chain during the inference process can be considered as a methodology for extracting additional knowledge in a domain in a semi-automatic way. It helps administrators to comprehend the effect of an event. If properly done, this reduces the number of wrong activities, and can even guide the refinement process. Moreover, this simulation mode can assist in finding out whether rules produce the same output as it would be produced by an expert in that domain.

To better illustrate this possibility, we consider the example shown in Figure 1. The value "*failure*" of the status of the resource "*application1*" indicates a problem in the domain modeled by this correlation engine (i.e. an Excel application is broken). However, the cause of a problem cannot be found, since there is no rule that can be evaluated (i.e. there are only rules related to Word applications). As the model acquisition is an ongoing process, the solution for that problem would be the extension of the model with a new rule that is needed. It is very difficult to find the right rule due to many of reasons: the model might contain million entities, the administrator might not have enough experiences, the existing rules might be ambiguous, etc.

The usage of ontologies as background knowledge can help identify knowledge gaps in the model. Let's now assume that the model of the correlation engine shown in Figure 1 is ontology-based. This background knowledge may be reused in two ways. Firstly, it may help find the rule that is most suitable for resolving a problem by taking into account the relationships defined into the corresponding ontology. For example, since the concepts "Text processor" and "Spreadsheet processor" are sibling concepts, they have many similarities. Therefore, the rules related to the "Text processor" (e.g. Word) might be probably satisfactory for the concept "Spreadsheet processor" (e.g. Excel) as well. Returning to the problem related to the "application1", an ontology-based correlation engine would resolve it by suggesting to consider RAM, since the "application1" is an instance of the concept "Spreadsheet processor" and the similarity between sibling concepts is reused (i.e. the rule Rule1 is evaluated).

Secondly, the administrator might be suggested to make some changes in the model that may yield this model better suited for the domain. For example by analyzing the usage data with respect to the ontology, the meaningful changes in the model can be discovered. If the rules related to the "Text processor" (e.g. Rule1) are often applied for resolving problems related to the "Spreadsheet processor", it might be useful to generalize them into "Office Tools" rules, since the concept "Office Tools" covers both concepts. In this case the rules Rule1 and Rule2 shown in Figure 1 should be updated.

Resources
application1.type="Excel"
application1.time=25min
application1.size=32MB
application1.status="failure"
...



Rule1:
If application.type="Word" and
application.time>20min and
application.size>30MB and
application.status="failure"
THEN there is a problem in RAM

Rule2:
If application.type="Word" and
application.time>20min and
application.lastAction>"save" and
application.status="failure"
THEN there is a HARD-DISK problem
...

Figure 1: An example for the gap analysis

4. Conclusion

In this paper we discuss how ontologies provide capabilities for automatic identification of problems in the domain knowledge processed by correlation engines. When such problems arise, ontologies assist the administrators in identifying the sources of the problem, in analyzing and defining solutions for resolving them.

Acknowledgements

The research presented in this paper was profited from fruitful discussion with members of the IBM Server Group in Boeblingen, Germany. It was financed by EU in the project "IST PROJECT 507237 - OntoGov".

5. References

[1] J. Kephart, D. Chess, The Vision of Autonomic Computing, *IEEE Computer*, January 2003., pp. 41-50.

[2] L.Stojanovic, et al., "The Role of Ontologies in Autonomic Computing Systems", *To appear in IBM Systems Journal*, Vol. 43, No. 3, 2004.

[3] IBM Tivoli System Automation for Linux on xSeries® and zSeries®, Tec. Report SC33-8210-00, 2002.