Kannisto P., Hästbacka D., Palmroth L. and Kuikka S. (2014).
Distributed Knowledge Management Architecture and Rule Based Reasoning for Mobile Machine Operator Performance Assessment.
In Proceedings of the 16th International Conference on Enterprise Information Systems, pages 440-449
DOI: 10.5220/0004870004400449
ISBN: 978-989-758-027-7
Copyright © SCITEPRESS
ICEIS 2014 - http://www.iceis.org

Distributed Knowledge Management Architecture and Rule Based Reasoning for Mobile Machine Operator Performance Assessment

Petri Kannisto[1], David Hästbacka[1], Lauri Palmroth[2] and Seppo Kuikka[1]
[1]Department of Automation Science and Engineering, Tampere University of Technology, P.O. Box 599, 33101 TAMPERE, Finland
[2]John Deere Forestry, P.O. Box 474, 33101 TAMPERE, Finland
{petri.kannisto, david.hastbacka, seppo.kuikka}@tut.fi, PalmrothLauri@JohnDeere.com

Keywords: Distributed Knowledge Management, Rule Based Reasoning, Operator Performance Assessment, Mobile Machines.

Abstract: The performance of mobile machine operators has a great impact on productivity that can be translated to, for example, wasted time or environmental concerns such as fuel consumption. In this paper, solutions for improving the assessment of mobile machine are studied. Usage data is gathered from machines and utilized to provide feedback for operators. The feedback is generated with rules that define in what way different measures indicate performance. The study contributes to developing an architecture to manage both data collection and inference rules. A prototype is created: rule knowledge is managed with decision tables from which machine-readable rules are generated. The rules are then distributed to application instances executed in various locations. The results of the prototype promote several benefits. Rules can be maintained independent of the actual assessment application, and they can also be distributed from a centrally managed source. In addition, no IT expertise is required for rule maintenance so the rule administrator can be a pure domain expert. The results bring the architecture towards a scalable cloud service that combines the benefits of both centralized knowledge and distributed data management.

1 INTRODUCTION

Even though the degree of automation in industries is constantly rising, various mobile machines still require a human operator, no matter how modern their equipment is. The difficulty of automatization is due to the irregular manner of not only operating environment but also the nature of work tasks. Despite the intelligence of human operators, they are prone to non-optimal operating of equipment.

Fortunately, as human beings have the ability to learn, systematic feedback may significantly improve production performance in the course of time. For instance, there could be potential to improve operating speed and reduce fuel consumption simultaneously. Considering that profit margins are narrow due to rising costs and a high degree of competition, even an improvement of a few percents could bring a substantial advantage.

The information and control system of a modern mobile machine is readily advanced which provides a basis for feedback generation. As there are numerous sensors around the machine that measure various magnitudes, a lot of data is available for collection.

The sensor data from the machine level can be automatically collected and stored by the manufacturer for example in the case a customer subscribes performance analysis and targeted maintenance and tuning for the machine fleet. From an enterprise wide storage, the data can be accessed for studying various indicators and comparing the values between machines of a similar type, age, working conditions, geographic area etc. Often, the information needs to be utilizable globally for analysis but also for generating performance reports based on e.g. individual machines, fleets, operators, or market areas.

However, the path from raw sensor data to a human-readable and analytic feedback report is long. Several questions arise. Which data is really relevant? Should all the data be collected just in case? Where to store the data? What is the methodology utilized to analyze the data to generate observations that form the basis of the feedback? Besides, there may be considerable differencies in machine utilization between the countries and market areas. Therefore, regional differentiation must be applied.

Among the several interesting aspects of operator assessment, this paper concentrates on two of them. The first is the *architecture* of the information management and assessment system. The second aspect is how to formulate and implement the actual *rules* applied in the assessment process. The rules define what level of performance is required for each measurement. The baseline for the architecture is how rules are accessed from distributed devices and how they are managed. The rules should be separated from the actual application logic so that rule modifications do not require any part of the assessment system to be recompiled. Also, no IT expertise should be required from a person to be able to create or maintain the rules.

The paper is organized as follows. Section 2 presents the methodology and background to this research as well as related work. In section 3 the problem is defined. The solution architecture and the knowledge management approach is explained in sections 4 and 5, respectively. Sections 6 and 7 contain discussion of results and conclusions.

2 RESEARCH CONTEXT

2.1 Research Methodology

The methodology applied in this work relies on constructive design science in which new artefacts, e.g. software, architectures, and algorithms, are developed to support projections of concepts. The developed solutions are then evaluated and compared to existing ones. Through a number of iterations, the new solutions finally provide new insights and better theories.

In this case the methodology was applied to design the conceptual architecture and the rule based knowledge management. The architecture was designed based on user requirements and developed to support the needs of distributed data acquisition, service-based operation, and rule-based knowledge management. Concerning the management of rule based knowledge, experiments with various rule modeling approaches were performed. Exploiting the results, the most appropriate solution was chosen and proposed. The performed requirements analysis sets the restrictions for the architecture and the solution proposed in this paper.

2.2 Rule-Based Reasoning

Even with simple rules, it might not be an easy task to implement a well-performing rule engine to apply reasoning rules to data. According to Schneier and Matignon, a challenge of applying rules is that even with a limited set of facts, the fact set will change once the rules are applied (Figure 1). An altered fact set means different circumstances so at least some of the rules must be executed again. The most straightforward way is to apply each rule after each fact insertion and start over whenever a rule is fired. However, the performance of this algorithm is slow with large data sets. (Schneier 2002; Matignon 2011)

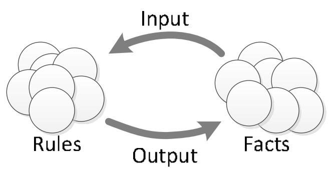


Figure 1: Facts are not only the input but also the output of rules. Based on (Schneier 2002; Matignon 2011).

The *Rete* algorithm developed by C.L. Forgy (1982) is a response to the performance challenge related to rule-based applications. It has two basic principles. First, a rule typically affects only a few facts, and second, rules may be structurally similar in that some conditions exist in more than one rule. Using these principles, a *graph* is built from the rules. A disadvantage of Rete is that it consumes relatively lot of memory. (Ingargiola) However, inference actions are fast because the graph enables targeting computation to where it is appropriate. (Schneier 2002; Ingargiola)

While Rete was obviously a significant step forward it has also received criticism, and several improvements have been developed. Miranker (1987) has proposed the *TREAT* algorithm which has been inspired by the shortcomings of Rete such as its memory consumption and some unnecessary computation it performs. After performing a comparison of Rete and TREAT, Wang & Hanson (1992) have stated that TREAT is typically faster but not always. Charles J. Forgy, the designer of Rete, has developed Rete II (RETE II) which showed a significant speed improvement in an experiment (Benchmarking CLIPS/R2). Further, the algorithm has been evolved into Rete-NT (Owen 2010). While the principles of the original Rete have been published, it seems that its successors Rete II and Rete-NT have been developed specifically for commercial systems so there is little information available about them. That is, while several articles discuss improving the original Rete, there is no information available how the suggestions align to the newest Rete versions. Articles suggesting extensions and improvements are, for instance, (Berstel 2002), (Ren & Wang 2008), (Xiao et al. 2009), (Liu et al. 2010) and (Yang et al. 2011). The Drools rule engine, for example, uses an optimized Rete version (Drools Expert User Guide 2012).

2.3 Related Work

While there are no publications with a topic similar to this, several studies have common aspects such as rules, distribution or knowledge management.

Chen & Wu (2003) introduce OKSA (Open Knowledge Server Architecture), a framework for managing knowledge in a network server. The framework contains service interfaces for retrieving semantic data as well as for rule-based reasoning.

Jung et al. (2006) take an information management approach. They propose an architecture combining knowledge management and business process management systems. Three types of information are stored: process template, process instance and process related. Knowledge storing and the business process point of view are relevant also in this study.

Rajasekar et al. (2006) have researched the application of rules for the management of data grids. While data grids are distributed by nature, it is challenging to make sure that the overall state of the data remains consistent. In that context, a rule-based approach is advantageous as it raises the flexibility of maintaining consistency constraints.

Marin-Perianu and Havinga (2007) have researched fuzzy logic based reasoning applied to wireless sensor networks. They have developed D-FLER, a rule-based fuzzy logic engine. Fuzzy output is generated from sensor readings which are then fed into an inference engine. The uncertainty of sensors is taken into account by processing the output of multiple neighbor sensors simultaneously.

Terfloth et al. (2007) have also performed rule-related research for wireless sensor networks. They have developed an architecture called FACTS which introduces a middleware layer to facilitate sensor-related programming by raising abstraction level. Rules are applied to events that arise from sensors.

Grobelny (2008) suggests a rule-based expert system to assist the composition of service architectures. The aim is to raise the abstraction level to enable composition for domain experts that are not specialized in software engineering.

Bontchev & Vassileva (2009) have studied the rule-based approach to create an adaptable e-learning environment. They have used the Drools rule engine to enable adaptation based on rules.

Dunkel et al. (2010) introduce an event-driven decision support system. In their paper, sensors provide data about the traffic situation in an area to facilitate traffic management. The data is refined to events that are processed by a rule engine. In the traffic management context, real-time constraints are present and data processing time is limited.

Nalepa et al. (2013) propose an architecture for business rules modeling. In their approach, the rules are modeled using BPMN (Business Process Model and Notation) diagrams. The approach provides control over in which order rules are applied. In the flow of a BPMN diagram, each task contains a set of rules to be executed. The approach aims for easier ruleset management; the visual representation makes a complex set of conditions more understandable.

3 PROBLEM DEFINITION

The development approach of this study follows agile principles. First, a use case analysis was performed; its results are described in this chapter. Second, the actual requirements of the architecture were formulated as given in chapter 4. Finally, a prototype was implemented as described in chapter 5. The process is analogous to an agile development process as explained by Douglass (2009): after a set of requirements has been specified, a prototype is implemented. As the prototype might not meet all the known requirements, any shortcomings will be taken into account in future development. The key is to control the development process by enabling proofing and experimenting design choices through a tangible implementation.

To begin the discovery of the problem, a use case analysis was performed with the objective to recognize actors that are involved in utilizing and maintaining the operator performance assessment system. This enabled defining requirements on the architecture and the information management.

Actors are involved in the assessment process in multiple roles; the most essential actors discovered in the analysis are illustrated in Figure 2.

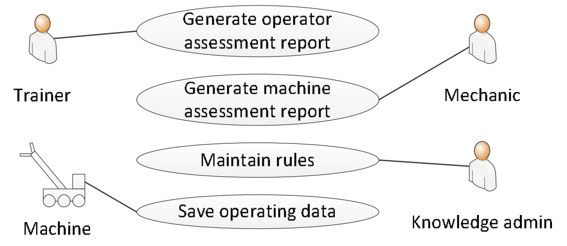


Figure 2: The most essential actors and the use cases performed by them that were identified.

Trainers generate feedback reports with the system. The feedback is then given to machine operators so that they can improve their way of working. A feedback report is simply a human-readable document that shows a comparison of the operator's performance to the average in the same market area during a certain time period. Typically, an operator does not even recognize that there is room for improvement before they receive the feedback. Naturally, a skilled operator will receive positive feedback, which indicates that they should keep working in a similar manner as before.

Mechanics generate machine reports that show equipment-related performance data. Modern mobile machines are often very complex with several subsystems so there may be multiple components that need maintenance, parameter tuning or calibration. These can be detected by investigating machine performance data. If a machine report indicates any need for service, the mechanic will inform the machine operator about it.

While mechanics and trainers are end-users, the *knowledge administrator* has an essential role in the actual assessment process. Based on their domain knowledge, they will create and maintain the rules applied to performance data. The knowledge administrator has qualitative domain expertise to determine which measures are relevant in the performance assessment. Whatever methodology is applied to administer the rules, the ultimate output of the knowledge administrator is the resulting report. Otherwise, the knowledge administrator is invisible to the end-users.

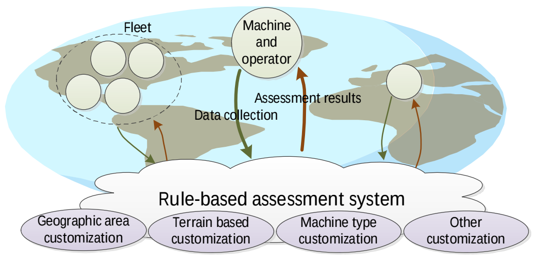
From the assessment system point of view, the *machine* itself is an actor as well. It gathers data from its sensors to be utilized for the assessment. In addition, the data should include machine type and the relevant setup that will have an effect in any forthcoming assessment.

Based on the identified actors and the use case analysis, the requirements can be summarized as follows. Whoever modifies the rules, no IT expertise should be required. That is, preferably no textual syntax should be utilized for rule definition, or at least the rule code should be generated based on some graphical syntax. Also, the rules should be stored in a location where they can be maintained separately from the actual application instances. The interface provided for rule maintenance should be accessible over the web. Finally, there should be a method to distribute an updated rule set, i.e. for generating the assessment reports, to distributed instances without the need to recompile the rules.

4 SOLUTION ARCHITECTURE REQUIREMENTS

Due to the requirements of the rule assessment system, architectural aspects are paramount in the development. While scalability and any other future requirements must be addressed, the improved architecture should also consider the potential benefits of a previously implemented assessment system and support a stepwise transition.

The concept of the assessment system is shown in Figure 3. Various machine types operate in different geographic areas, and data can be collected for individual machines as well as an entire machine fleet of a given entrepreneur. After each cycle of operation, the collected data is submitted to the assessment system. The collected data may be utilized for generating a reference dataset for the machine type. After the data submission, the reference dataset is compared to the data collected from operation and the results are shown to the operator. The salient aspects of the architecture are explained in the following sections.


Fleet
Machine and operator
Assessment results
Data collection
Rule-based assessment system
Geographic area customization
Terrain based customization
Machine type customization
Other customization
Figure 3: The architecture needs to support data collection from machines all over the world, provide means for customization of the assessments based on different criteria, and deliver the assessments reports based on centrally managed knowledge.

4.1 Data Acquisition and Storage

Modern mobile machines have sophisticated sensors that enable versatile application development.

Originally, the sensors have been introduced to serve the needs of constantly developing machine automation. Today, they provide new value by providing the basis of the data collection driven by assessment needs.

Despite the possibilities of modern wireless networking, an off-highway machine cannot be assumed to have a persistent and stable Internet connection. Mobile Internet is not available everywhere and no network is fully reliable. Also, whichever is the technology utilized to receive and store the data, occasional downtime is probably inevitable. Thus, it is required that a machine can store measurement data locally – if not for long periods then at least for temporary caching.

Finally, all the data gathered from machines should be stored remotely for future utilization. To enable consistent data management easily, a common storage must be used by all the machines.

To reach the required flexibility, the architecture should be designed to function on a cloud platform. Obviously, it would enable both global knowledge management and storage as well as local customizations. Moreover, cloud services are scalable, and as there is no single point of failure, the availability of the service would be improved.

4.2 Dispersed Information Processing

The measured performance data is processed on the machine level as well as for groups of machines at the central data storage. The processing and analysis of the data differs depending on the purpose for which the information is produced. The process is explained in more detail by Palmroth (2011).

On the machine level, the distributed control system of the machine and sensors produce large amounts of data. This data must be pre-processed such that it contains the relevant information needed for the performance monitoring and assessment purposes. The data must also be compressed in order to transfer it from the machine. Naturally, data compression is always a tradeoff between fidelity and the amount of data to be transferred.

Quantitative knowledge of the rule based system is based on statistics of performance data measured from large machine fleets on separate geographic and market areas. This data is stored in the central data storage. It is a very important concept in the knowledge management architecture that only qualitative domain knowledge is required from the *knowledge administrator*. It is not necessary to master all the numerical values of performance measurements from different types of machines etc., because the quantitative information is produced by the statistics of measured performance data.

4.3 Decentralized Knowledge Management

Domain knowledge has multiple dimensions. Some qualitative knowledge may be relevant only for a specific machine type or a set of machine types with a specific equipment setup. In addition, there could be considerable differences in machine utilization between different geographical areas, work methods, operating temperatures and terrain types. On the other hand, some knowledge may also be considered general regardless of areas and machine types. When there are relevant differences, they should be incorporated in the domain knowledge.

There is also a lot of variation between operating locations. The terrain may vary from flat to hilly or even mountainous. The flatter the land the easier and less resource-consumptive operation should be expected. Also, the surface of the land has an effect as well: it is easier for any machine to move on hard land. Any obstacles such as plants or stones will also make operating more challenging. Finally, there may be variation between the load or material types processed by the machine: some of them may require more work for processing than the others.

One of the most essential questions of knowledge management is the global and distributed nature of the knowledge and the need to be able to access it from various locations. With increasing amounts of data constantly accumulating from machines all over the world, typical big data challenges are expected to become important. The central storage will make it straightforward to access the data; however, as the knowledge actually originates from several locations, it will be challenging to store it in a single-format storage. That is, to reach an architecture that is both easy to manage and also adaptable to meet the requirements of different areas, careful design is required.

In this work, a high level of centralization is required to make it possible to manage the rule knowledge. Due to the level of domain expertise required to manage the rules, there will only be a few professionals that have the appropriate skills. That is, by utilizing a centralized architecture, it is possible for the professionals to have a control of the rule knowledge as a whole.

4.4 Assessment as a Service

Operator assessment can be seen as a service provided by the machine manufacturer. In general, the data is owned by the machine owner but delivered to the service provider per service agreement. The customers will order reports to improve their efficiency and effectiveness; however, extra revenue will also be generated for the manufacturer. To ensure that the operator assessment system reflects the business requirements, the system should be implemented utilizing service thinking.

Compared to a monolithic software application, service-based design performs better in distributed business. Maintaining and updating a software application is easier if instances are not run at customers' locations but rather in an environment controlled by the service provider. Also, principles such as loose coupling between applications, a high abstraction level of interfaces and aiming at reflecting business requirements in design are likely to ease both development and maintenance.

On the customer side, it is clear that some client application is required. Due to the capabilities of the web browsers and the web developers of today, the client application could actually be a web browser that accesses a website for report generation.

5 RULE-BASED PERFORMANCE ASSESSMENT PROTOTYPE

5.1 Implementation

To bring the current assessment system closer to the above concept, a prototype was implemented. Its general architecture is illustrated in Figure 4. The rule base has been implemented with the open source Drools framework which provides a browser interface for maintaining the rule knowledge. The analysis software accesses the rules and applies them to the machine data to make the assessments. The prototype has been developed for forestry domain but can be generalized to any mobile machine type.

Figure 5 presents a more detailed description of the implementation elaborating the transformation of rule-based knowledge. For this, there are three major components: data processor, rule base and rule converter.

The data processor application is the most important component. It generates performance reports according to the rules. The core of the application, the data processing logic, has been implemented in Matlab and compiled as a desktop application. The calculation functionality provided by Matlab is versatile compared to general-purpose programming languages. In the prototype, it is implemented as a desktop application that caches the rules from the rule base.

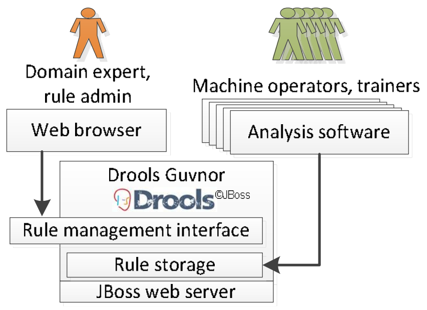


Figure 4: The prototype implemented using the Drools framework includes an interface for the domain expert for managing the rule knowledge that is used by the analysis software making the assessments.

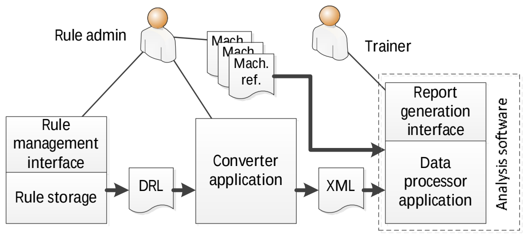


Figure 5: Rule based domain knowledge is transformed for the Matlab based analysis software.

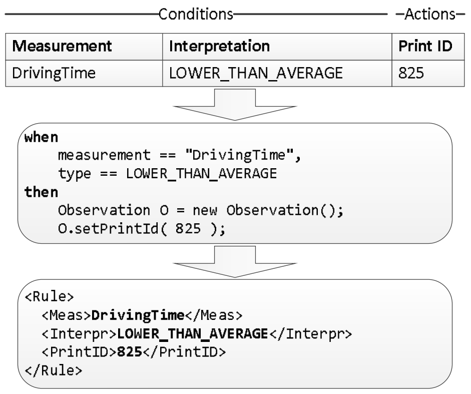
The rule base has been built in a web server utilizing the Drools rule framework. Drools may be included as a library to any Java application, but the rules may also be managed and executed in a web server application called *Guvnor*. Guvnor has a graphical browser interface that is easily accessible over the Internet. The rule base provides also a rule engine applying the RETE algorithm for execution.

Drools supports various means for defining rules such as direct input in a rule language and decision tables. The rule language, DRL (Drools Rule Language) (see Drools Expert User Guide 2012), covers rule needs rather widely. However, while it may be intuitive for a software engineer, it is certainly not understandable for a person unfamiliar with programming. Also, if the number of rules is large, it may be difficult to manage the rules as a whole if all of them are in textual form.

To eliminate the burden of learning the DRL syntax, it is possible to develop domain specific languages (DSL) over DRL. However, as a DSL can still lack a graphical representation, *decision tables* may also be utilized to model the rules. In a decision table, each row defines a rule, and each column represents the value of a condition. The trade-off of decision tables is their limited power of expression compared to DRL, and they also require that the rules within one table are structurally similar.

In this study, decision tables are utilized due to their intuitive and easy-to-learn nature. Any limitations related to the power of expression are not important as the rules are rather similar by their structure. Once any decision tables have been created, the rules defined in them are downloadable as a DRL file. The rule engine component included in Guvnor is not utilized as applying inference locally will raise the flexibility of the solution in terms of whether there is a persistent Internet connection or not.

The third component in the architecture is a converter application that enables the integration of the rule base and the data processor. The data processor receives rules in a proprietary XML format while the output of the rule base is DRL so a conversion is required between them. The power of expression provided by DRL is rather extensive, but only a subset of it is required by the data processor. That is, only certain structures of DRL are recognized and converted to the XML format. As long as simple decision tables are used, no compatibility problems are expected as the DRL code generated from decision tables is uniform. The conversion of rule formats is illustrated in Figure 6.



Measurement	Interpretation	Print ID
DrivingTime	LOWER_THAN_AVERAGE	825

```
when
    measurement == "DrivingTime",
    type == LOWER_THAN_AVERAGE
then
    Observation O = new Observation();
    O.setPrintId( 825 );
```

```
<Rule>
  <Meas>DrivingTime</Meas>
  <Interpr>LOWER_THAN_AVERAGE</Interpr>
  <PrintID>825</PrintID>
</Rule>
```

Figure 6: A simplified example of representing a decision table row as DRL and converting it to an XML node.

5.2 Example of Use

To clarify the proposal in this paper, this section provides an example of use of the prototype in mechanized forestry. The example is about forwarder operations in Brazil and in Denmark. Forwarder is a forest machine type that is used to collect felled trees from forest and to transport them to roadside for further transportation.

The entire performance assessment process is based on the rules created by knowledge administrators. Rules are created separately for different machine types such as harvesters or forwarders, including the various models offered by the manufacturer. The knowledge administrators with global expertise create the global rules that form the basis of all the rules of a machine model. To maintain the rules, they log into Drools Guvnor using a web browser. The rules are defined as decision tables using a graphical user interface.

Both Brazil and Denmark have their local characteristics that are considered by the regional knowledge administrators of the countries, e.g. an area sales support manager with the required domain knowledge. For example, Denmark is a relatively flat country which makes driving the forwarder fast and easy. In contrast, there are hilly forests in Brazil which makes driving more demanding, raising fuel consumption and lengthening driving times. On the other hand, the trees in Brazil are almost entirely eucalyptus that is easy to collect, whereas Denmark has a variety of different tree species which need to be sorted separately. Therefore, longer loading times are expected in Denmark. Both Danish and Brazilian knowledge administrators make local customizations to the global rules. Like global knowledge administrators, the local ones maintain rules as decision tables with Drools Guvnor.

After local customizations, the assessment rules are ready for distribution. In the prototype, each local knowledge administrator downloads a DRL representation of the decision tables manually from Drools Guvnor. Then, with the converter application, the DRL representation is converted to XML so it can be utilized by the analysis software. Finally, the XML rule sets are distributed to trainers.

Once customized, transformed and distributed, the rules can be utilized in assessment. The Brazilian and Danish operators use the forwarders in forests. After some time period, trainers retrieve raw usage data from each machine with their laptops and feed the data to the analysis software. Once the analysis software has generated performance data from the raw data, it applies the localized XML rule set to the performance values. As the result, a textual feedback document is generated for a specific operator and a specific time period. It is the final output of the process; the results are given to operators so they can improve their way of working.

As the understanding of performance assessment improves over time, the assessment rules are likely to evolve. This will require modifications in not only global but also in local rule sets. Whenever the global rule set is modified, it will be communicated to local admins that consider the modifications.

To conclude, the assessment process is as follows:

1. Global knowledge admin: create global rules
2. Local knowledge admin: localize rules
3. Local knowledge admin: convert and distribute rules
4. Trainer: get raw data from machine and run assessment with the analysis software
5. Operator: learn from assessment results

6 RESULTS AND DISCUSSION

During the evaluation of the prototype, it was found out that the chosen approach has several benefits. Whenever rules are applied in the data processor, no Internet connectivity is required which is often the case with off-highway vehicles. The rule-based knowledge can also be centrally managed by a domain expert without ICT skills. The architecture is also relatively straightforward to implement.

In the prototype, however, manual work is still required to distribute rules – whenever the rules are changed, a new instance of them must be made available to the analysis application instances. In addition, having a separate conversion application adds an extra component and a new level of complexity – the entire distribution process should be made automatic to maximize the ease of use and management. However, even though complexity is not desirable, a multistep conversion process makes it possible to change one step independently as long as interfaces remain as they are. The ultimate goal is to cache the entire rule distribution service for the analysis software: if there is an Internet connection, check for recent changes, if not, utilize a cached version.

From the point of view of this work, Drools has both advantages and disadvantages. Its capabilities for rule modeling are well beyond the complexity required for the current system. So, there is no shortcoming from this point of view, but Drools also makes it possible to model rules so complex that they cannot be converted for utilization in analysis. However, a support for rules more complex than the current ones might be required in the future. In addition, the current decision table format sets an effective restriction to rule complexity so there is little danger that "too complex" rules are modeled. On the other hand, a more flexible modeling method than decision tables might be needed in the future – any other methods provided by the current Drools require too much expertise for most non-ICT users. One advantage of Drools is its wide utilization in both research and practical applications so further development and support for the framework are expected. To conclude, while Drools is not a perfect solution, its features provide a solid foundation.

A global service-oriented approach would, perhaps, suit better for the requirements than running multiple instances of the application in various locations. The approach of the prototype allows for realizing the concept to provide the operator assessment also as a centralized service over the Internet, as envisioned in section 4.4. The performance assessment system should, however, be later implemented as a cloud service to enable not only centralized management but also distributed one where it is necessary. As centralization and distribution can in general be considered conflicting goals, it is an important design question which functions should be centralized and which ones distributed or local.

The current requirement to have a trainer on-site to get raw machine data and generate feedback for operators is a limitation. In the vision, raw machine data would be stored in the cloud. Then, operators could utilize the analysis service to get feedback whenever they want to. Of course, the expertise of a trainer could still be beneficial for an operator when interpreting feedback.

7 CONCLUSIONS

The paper introduces a concept of a distributed knowledge management solution. A prototype has been developed for maintaining rule-based domain knowledge that is used in operator performance assessment. The developed functionality allows domain experts to maintain the knowledge without ICT skills. A system architecture has also been developed that supports centralized management and customization of this knowledge and the globally distributed utilization when generating assessment reports. The solution has been applied in the assessment of forest machine operators' work technique and performance.

The implemented rule management system enables rule modeling and distribution for non-ICT users. However, there is room for improvement in ease of modelling as well as adaptability and flexibility – simply put, how to have control over the entire assessment process with less human effort.

While rule-based architectures and distributed knowledge management systems have been implemented before, this work is novel in the domain of mobile machines. In addition, the support for not only global distribution and management but also the consideration of local customizations is an advance in this field of science. Moreover, different usage roles are also considered in this paper. The approach can be adapted to other applications for mobile machines or settings where similar information management challenges are present.

In the future, more development work is needed to improve flexibility, scalability and reachability of the system. Transition towards a cloud based approach could improve this by facilitating central management of shared resources as well as utilization and integration of data sources, i.e. global machine data and machine fleets. Another interesting goal is applying the approach towards condition monitoring and diagnostics for detecting events from sensor data for reactive or preventive maintenance.

REFERENCES

Benchmarking CLIPS/R2. Production Systems Technologies, Inc. Available at: http://www.pst.com/benchcr2.htm (Last visited 9th Oct. 2013)

Berstel, B., 2002. Extending the RETE Algorithm for Event Management. In *TIME 2002, Proceedings of the Ninth International Symposium on Temporal Representation and Reasoning*. IEEE. pp. 49-51.

Bontchev, B., Vassileva, D., 2009. Rule-driven approach for adaptable e-learning content delivery. In *EISTA 2009, Proceedings of 7th International Conference on Education and Inf. Systems, Technologies and Applications*. IEEE. Pp. 10-13.

Chen, H., Wu, Z., 2003. OKSA: an open knowledge service architecture for building large scale knowledge system in semantic Web. In *International Conference on Systems, Man and Cybernetics*. IEEE. Vol. 5, pp. 4858-4863.

Douglass, D., 2009. *Real-Time Agility*, Addison-Wesley. 1st edition.

Drools Expert User Guide, 2012. Available at: http://docs.jboss.org/drools/release/5.5.0.Final/drools-expert-docs/html_single/

Drools Guvnor. Available at: http://www.jboss.org/drools/drools-guvnor.html (Last visited 9th Oct. 2013)

Dunkel, J., Fernández, A., Ortiz, R., Ossowski, S., 2011. Event-driven architecture for decision support in traffic management systems. In *Expert Systems with Applications*, 38(6), 6530-6539.

Forgy, C.L., 1982. Rete: A fast algorithm for the many pattern/many object pattern match problem. In *Artificial Intelligence*, Volume 19, Issue 1, September 1982, Pages 17-37, DOI: 10.1016/0004-3702(82)90020-0.

Grobelny, P., 2008. Knowledge representation in services oriented architecture. In *Przeglad Telekomunikacyjny*, 6, 793-796.

Ingargiola, G. The RETE Algorithm. Available at: http://www.cis.temple.edu/~giorgio/cis587/readings/rete.html (Last visited 8th Oct. 2013)

Jung, J., Choi, I., Song, M., 2006. An integration architecture for knowledge management systems and business process management systems. In *Computers in Industry*, 58(1), pp. 21-34.

Liu, D., Gu, T., Xue, J.-P., 2010. Rule Engine Based on Improvement Rete Algorithm. In *ICACIA 2010, International Conference on Apperceiving Computing and Intelligence Analysis*. IEEE. pp. 346-349.

Marin-Perianu, M., Havinga, P., 2007. D-FLER – a distributed fuzzy logic engine for rule-based wireless sensor networks. In *Ubiquitous Computing Systems* (pp. 86-101). Springer Berlin Heidelberg.

Matignon, C.-A., 2011. Rete Algorithm Demystified! — Part 2. Available at: http://techondec.wordpress.com/2011/03/14/rete-algorithm-demystified-part-2/

Miranker, D.P., 1987. TREAT: A Better Match Algorithm for AI Production Systems; Long Version. Technical report AI TR87-58.

Nalepa, G. J., Kluza, K., Kaczor, K., 2013. Proposal of an Inference Engine Architecture for Business Rules and Processes. In *Artificial Intelligence and Soft Computing* (pp. 453-464). Springer Berlin Heidelberg.

Owen, J., 2010. World's fastest rules engine. InfoWorld. Available at: http://www.infoworld.com/t/business-rule-management-systems/worlds-fastest-rules-engine-822

Palmroth, L., 2011. *Performance Monitoring and Operator Assistance Systems in Mobile Machines*. Doctoral dissertation, Department of Automation Science and Engineering, Tampere University of Technology, Tampere, Finland.

Rajasekar, A., Wan, M., Moore, R., Schroeder, W., 2006. A prototype rule-based distributed data management system. In *HPDC workshop on Next Generation Distributed Data Management* (Vol. 102).

Ren, Z., Wang, D., 2008. The Improvement Research on Rule Matching Algorithm Rete in Electronic Commerce Application Systems. In *WiCOM 2008, 4th International Conference on Wireless Communications, Networking and Mobile Computing*. IEEE. pp. 1-4.

RETE II. Production Systems Technologies, Inc. Available at: http://www.pst.com/rete2.htm (Last visited 9th Oct. 2013)

Schneier, B., 2002. The Rete Matching Algorithm. Available at: http://www.drdobbs.com/architecture-and-design/the-rete-matching-algorithm/184405218

Terfloth, K., Wittenburg, G., Schiller, J., 2006. FACTS – a rule-based middleware architecture for wireless sensor networks. In *Comsware 2006, First International Conference on Communication System Software and Middleware*. Pp. 1-8. IEEE.

Wang, Y.-W., Hanson, E.N., 1992. A Performance Comparison of the Rete and TREAT Algorithms for Testing Database Rule Conditions. In *ICDE 1992, Eighth International Conference on Data Engineering*. IEEE. pp. 88-97.

Xiao, D., Tong, Y., Yang, H., Cao, M., 2009. The Improvement for Rete Algorithm. In *ICISE 2009, 1st International Conference on Information Science and Engineering*. IEEE. Pp. 5222-5225.

Yang, P., Yang, Y., Wang, N., 2011. IRETE: An Improved RETE Multi-entity Match Algorithm. In *ICECC 2011, International Conference on Electronics, Communications and Control*. IEEE. Pp. 4363-4366.