Enabling Centralised Management of Local Sensor Data Refinement in Machine Fleets

Petri Kannisto and David Hästbacka
Department of Automation Science and Engineering, Tampere University of Technology,
P.O. Box 692, FI-33101 Tampere, Finland

Keywords: Distributed Knowledge Management, Mobile Machinery.

Abstract: In modern mobile machines, a lot of measurement data is available to generate information about machine performance. Exploiting it locally in machines would enable optimising their operation and, thus, yield competitive advantage and reduce environmental load due to reduced emissions. However, optimisation requires extensive knowledge about machine performance and characteristics in various conditions. As physical machines may be located geographically far from each other, the management of ever evolving knowledge is challenging. This study introduces a software concept to enable centralised management of data refinement performed locally in the machines of a geographically distributed fleet. It facilitates data utilisation in end user applications that provide useful information for operators in the field. Whatever the further data analysis requirements are, multiple preprocessing tasks are performed: it enables outlier limit configuration, the calculation of derived variables, data set categorisation and context recognition. A functional prototype has been implemented for the refinement of real operational data collected from forestry machines. The results show that the concept has considerable potential to bring added value for enterprises due to improved possibilities in managing data utilisation.

1 INTRODUCTION

The current era of industrial informatics has brought ever developing intelligent devices, data processing methods and sensor technology. Additional value can be gained from existing devices by collecting data and analysing it to have new information and knowledge. The importance of data analysis has been emphasised not only for business in general (LaValle et al., 2011) but also in industrial context (Duan and Xu, 2012). For production, this also applies to mobile machines such as earthmoving, mining or forestry. Performance improvements not only bring competitive advantage but they also save resources and reduce emissions to the environment.

In this paper, a software concept is introduced – intended for service architectures – to enable centralised management of fleetwide sensor data refinement which is performed locally in mobile machines. The operation of modern machinery typically requires a high level of expertise, and even a skilled operator rarely has the technological knowledge required for optimal operation. That is, various feedback applications should be utilised to improve performance. In the data measured during operation, a lot of implicit information is available not only about the machine itself but also the material or the goods being processed. In an ecosystem, the number of machines and the amount of data can be arbitrarily large, and the machines may be geographically distributed. A centrally managed data refinement solution facilitates using all the potential of data as it unifies the information available for actual end user applications in various machines. The applications may, for instance, provide assistance in machine operation or adjustment. As data processing expertise and requirements are likely to evolve, frequent updates are expected.

The practical contribution of this work is the implementation of an externally manageable intermediary component that refines local machine data. It accomplishes essential first-hand tasks thus generating information and facilitating further analysis. The component can be customised externally by delivering methods and configuration data that define the local refinement process. As several machines can receive an identical refinement configuration set, centralised management is enabled.

Kannisto, P. and Hästbacka, D.
Enabling Centralised Management of Local Sensor Data Refinement in Machine Fleets.
DOI: 10.5220/0006045600210030
In *Proceedings of the 8th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2016) - Volume 3: KMIS*, pages 21-30
ISBN: 978-989-758-203-5

The structure of this paper is as follows. Related work is discussed in Section 2. Section 3 discusses the actual problem followed by a solution in Section 4. A forestry machine related prototype implementation is introduced in Section 5. Section 6 presents the results while Section 7 concludes the paper.

2 RELATED WORK

Among the publications in the industrial domain, no corresponding points of view have been found so this work is considered novel. None of the discovered previous studies address a similar data processing workflow with the configurability aspect and a similar level of detail. That is, this section summarises the work related to either machine data refinement, equipment data exchange or context awareness as each aspect is included in this paper.

Farming equipment related data collection or exchange has been researched in various papers. In (Steinberger et al., 2009), farming equipment data is exposed in a service architecture. In (Iftikhar and Pedersen, 2011), device data is exchanged bidirectionally between office computers and farming machines. In (Peets et al., 2012), there is a solution for data collection from various types of sensors. In (Fountas et al., 2015), an information system concept is introduced for the management of farming machines.

There are also other publications related to mobile machine data processing. In (Palmroth, 2011), the analysis of mobile machine data to assist operator learning is covered. A knowledge management solution for operator performance assessment in the field is considered in (Kannisto et al., 2014). In (Kannisto et al., 2015), a service architecture is introduced to manage the information and knowledge required to assist machine parameter optimisation locally in machines. All of these studies contain machine data refinement, and the latter two have an information system architecture aspect. However, none of them has a similar level of detail in configurability.

Fault diagnostics and condition monitoring are also related as they consider generating information by processing measured data. Various mathematical methods can be utilised for diagnostics as presented in (Yang and Kim, 2006; Basir and Yuan, 2007; Banerjee and Das, 2012). Condition-based maintenance (CBM) is enabled by utilising collected condition data (Jardine et al., 2006). Recently, even wireless sensor networks (WSN) have been utilised in diagnostics (Lu and Gungor, 2009; Hou and Bergmann, 2012). These studies focus on data processing rather than knowledge management essential in this work.

Context recognition has been researched for a long time, and various methods as well as applications have been suggested. In (Khot et al., 2006), there is a mathematical approach to recognising the context and the position of a tree planting robot; position information from various sources is combined mathematically to reduce error. Machinery is the domain also in (Golparvar-Fard et al., 2013) where earthmoving equipment actions are recognised from video. Human activities recognition has also been researched including hospital work (Favela et al., 2007), car manufacturing (Stiefmeier et al., 2008) and general activities (Choudhury et al., 2008). In this paper, relatively little weight is put on context recognition so the method should not be compared with the advanced context recognition methods found in literature.

3 DATA PROCESSING NEEDS FOR MACHINE FLEETS

This paper introduces a software concept to centrally manage data refinement performed locally in the machines of a geographically distributed fleet. The idea is to process machine data to facilitate further utilisation, especially for instant local use. A considerable challenge stems from the errors that real-life machine data typically contains: there is a need to apply domain expertise by specifying conditions and limits that determine which measurement values should be considered valid. For instance, measurements are never completely accurate, and for one reason or another, a sensor may either systematically or randomly give erroneous output. The physical world rarely acts ideally. Another essential requirement is context recognition and consideration – this need comes from the argument that some domain knowledge is context dependent. Obviously, the operating environment and the type of work being performed largely affect what kind of measures and performance are expected. Further requirements are data set categorisation and the calculation of derived indirect values as they may provide valuable information.

Scalable configurability is essential: it must be possible to control data refinement even after it has been taken into use in a large geographically distributed machine fleet. Figure 1 illustrates this. Machine data collection enables fleetwide knowledge generation within the enterprise. The knowledge is translated to a data refinement configuration to be delivered to machines. In each machine, the configuration is utilised in data generation for analysis applications that generate added value such as feedback about machine operation.

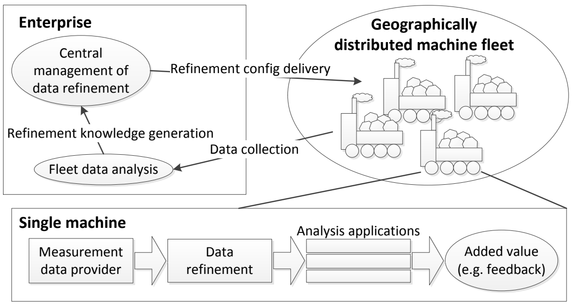

Figure 1: Data refinement configuration is delivered to machines so the refinement can be executed locally for analysis applications.

In this work, data is structured as *data item collections*. A data item collection contains measurement and parameter values saved at a certain point in time thus providing a snapshot of machine state and performance. Data items are stored as a set of key-value pairs that enable access to data items using their identifiers. It is assumed that the machines of the same type have an identical key set in their data item collections. Once a data item collection has been retrieved, its items can be utilised for calculating or inferring derived data and information or to resolve prevailing context.

To clarify the concept, let us consider an example in the forestry domain. As tree stems are processed with modern equipment, a lot of measured stem data is available such as the felling diameter, length or how quickly the stem has been processed with the machine. For each processed stem, the related measurements will be stored in a data item collection so the stems can be processed one by one with all related data available.

Machine type specific data item collection processing is likely required. First, there is expected to be variation in available measurements between machine types. For instance, as the degree of automation in tree stem processing keeps improving, a new machine model likely has more measurement items available compared to old ones. Second, models likely have variation in productivity, fuel consumption and other performance values. Third, variation in machine parametrisation is also expected due to differences in components such as hydraulic valves that control the machine boom. Parameter sets may vary as well as how a certain parameter affects machine operation.

Naturally, there are various reasons why a measurement may fail. Even a modern sensor may lack the ability to indicate if it has succeeded in measuring a value or not. Even if a sensor were not malfunctioning, there is still a possibility that its reading is not reliable – for instance, the sensor might have come off its installation position thus measuring something unexpected. In any case, it must be considered if each measurement value is reasonable or not. The motivation of outlier consideration is discussed, for instance, in (Osborne and Overbay, 2004).

As data item collections are persisted for later utilisation, each measurement value should be stored as such not to eliminate the possibility to recalculate values. This applies especially to cases where long-time historical data is required in analysis. If a measurement value is considered out of outlier limits and automatically declared a failure, it will be impossible to reprocess it in case of a later change in outlier conditions. Therefore, in many cases, it is a good practise *not* to store any values calculated from measurements as calculation formulas might evolve. Naturally, in some applications, if original values are not needed for sure, it may also be appropriate to save storage space by only saving the essential derived values rather than all raw values.

To run data analyses in a large scale, it is beneficial if data item collections are categorised. There may be considerable systematic variation in their values. Not to treat them as a homogeneous mass (what they certainly are not), at least rough categorisation is beneficial so each data item collection may be treated within an appropriate group. In forestry, each stem may be categorised after its size or tree species as it likely affects productivity – if the processing of large trees is being optimised, little trees should be ignored. As categorisation is performed based on measured values, it is subject to failures; it cannot be performed if some required value has been measured incorrectly.

Mobile machines may operate in varying environments so the power of context awareness should be exploited as the context may significantly affect how a machine can perform (Väyrynen et al., 2015). Depending on the context, an absolute numeric value may be relatively high or low. It must be considered if performance value comparison is appropriate if the values have been measured in different contexts. For instance, performance is likely low in unfavourable conditions: the temperature may affect fuel consumption, rough terrain makes machine movement slower and so forth. In context classification, its subtleness and other aspects must be considered depending on the application area. Another important consideration is knowledge evolution: it may also be required to update the selected context classification method sometimes.

Context recognition is essential also in forestry. Even inside a relatively small geographical region, there may be a lot of variation between forests: the type of land may affect machine performance, and tree species may also vary. Also, the type of work being performed (final felling, thinning or other) always affects absolute productivity values.

4 EXTERNALLY MANAGEABLE DATA REFINEMENT

4.1 Workflow

Considering given requirements, a solution can be designed. The flow of the application run locally in machines is illustrated in Figure 2. There are four main phases complemented by context consideration. To enable the utilisation of constantly evolving domain expertise, some phases utilise externally defined methods or configuration files. Each phase is explained in the coming paragraphs.

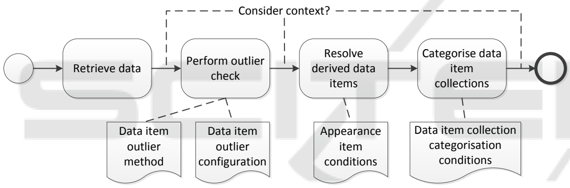

Figure 2: Data refinement flow.

First, measurement values are retrieved; they are stored in data item collections realised as key-value pairs. For a certain machine type, each collection is expected to have the same key-value pairs. In forestry, a reasonable data structure is to have a data item collection for each processed tree stem.

Then, an outlier check is performed. Whatever the utilised method is, it should be applied early as it may affect forthcoming data processing.

In the next phase, any derived variables are resolved. Often, not all desired variables can be directly measured so calculation may be required. Naturally, a derived variable cannot be calculated if any required measurement has failed. In this work, derived Boolean values associated to a data item collection are called *appearance* items: whether some condition set is fulfilled by the collection or not. For instance, in forestry data, an appearance item may express whether a stem is a spruce. The information may be utilised in further data analysis to determine which stems are considered interesting – occasional birches in a spruce forest may not be interesting.

Finally, each data item collection is categorised. Whatever the categorisation criteria are, technically, they consist of condition sets on measurement values. If a data item collection has a failed measurement value that is required for categorisation, the collection is ignored in tasks where categories are essential.

Depending on the application, context awareness may be applied in several phases. Context information may even affect the outlier check; for instance, it may determine which numeric outlier limits are applied or it may determine what kind of outlier check method is utilised. Later in the refinement flow, the context may affect how derived data items are resolved. However, some context awareness methods may require data item collection categorisation results so they cannot be utilised earlier. In the end, even though the workflow has a certain phase set, its design is adaptable in terms of context awareness.

Let us consider the forestry example again. First, an outlier check is required. For instance, if a measured value is beyond its reasonable limits, it must be declared a failure. Second, derived variables are calculated. Typical effectiveness variables (such as wood volume productivity while processing a single stem) are such as they cannot be measured directly. Also, some derived variables may require considering multiple data item collections (i.e. stems; such as the mass of processed wood per working hour during a day). Another derived variable could be the Boolean value (i.e. appearance item) whether a stem is "large" which involves the comparison of its felling diameter to a specific limit. Third, data item collections are categorised according to predefined conditions. Depending on the objective of the categorisation, stem categories could include tree species, tree sizes or both. Besides the mentioned phases, context-awareness may be applied in multiple parts in the flow. One option is simply to let the predominant tree stem category determine the prevailing context – this depends on the implementation of the application.

4.2 Configuration Management

A software component has been designed to implement the application flow that utilises externally provided configuration documents (see Figure 3). In each individual machine, the component retrieves data from the machine information system and refines it for further analyses. The number of machines executing the flow is arbitrary as well as their geographic locations compared with each other and the enterprise office. Several new specification requirements arise from configuration: the payload enclosed in each document, their format, and the way the configuration is delivered to machines. Whichever is the way these requirements are met, a local configuration cache is likely required as a machine may operate long periods of time without a connection to any centralised configuration storage.

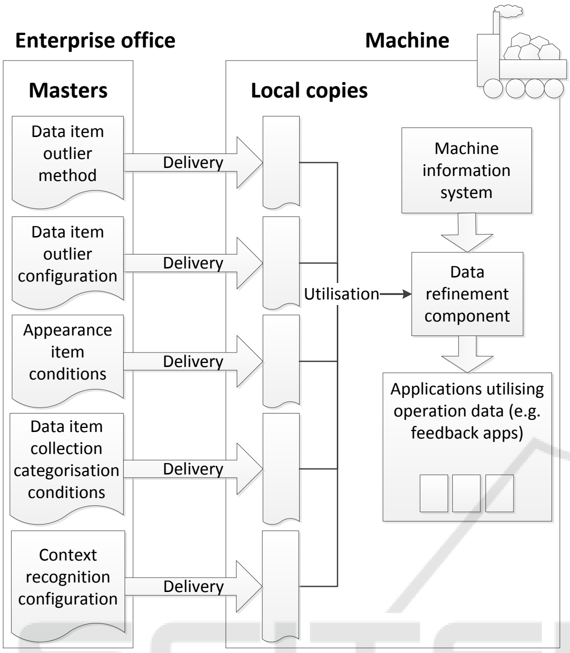

Figure 3: Configuration management illustrated.

The modern information technology portfolio provides several options to deliver configuration documents from the office environment to mobile machines. Even widely used standard Internet technologies can be utilised here: for instance, HTTP (Hypertext Transfer Protocol) is suitable for transfering text-based documents to physical machines. In that case, a network connection (such as the Internet) is naturally required. Be there no network, the configuration may be delivered using another technology (even by copying files manually); for instance, regular machine maintenance could be an occasion suitable for delivery. Finally, whatever the configuration delivery method is, regular updates should be enabled so each machine has as up-to-date a configuration as possible.

5 CONFIGURABLE DATA REFINEMENT PROTOTYPE

5.1 Implementation

Following the specified concept, a prototype has been implemented for tree stem data processing in the forestry domain. Configuration delivery from the enterprise office to machines is left as a future task. Yet the implemented component prototype is configurable, and its design allows configuration delivery in whichever way suits best for the use case. The prototype implementation is not supposed to be a fully comprehensive solution – it rather illustrates that the concept is functional in general.

The data item collection refinement flow in the prototype is illustrated in Figure 4. There will be a data item collection for each processed tree stem and the logs made from it. First, measurement values are retrieved and structured as data item collections. Then, an outlier check is performed for each measurement value in each data item collection; the data items that do not match their conditions are marked as failed. Next, appearance items are resolved by checking whether each data item collection satisfies each appearance condition set or not. Finally, stem data item collections are categorised based on their values. Here, it must be noted that if some measurement value required for categorisation has failed (per outlier check), the category cannot be resolved. Instead, the stem data item collection (and the related log data item collections) will not be further processed.

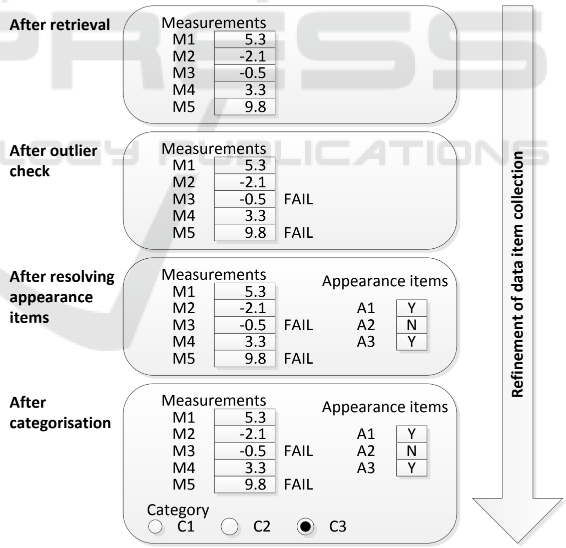

Figure 4: Data refinement in the prototype implementation.

The method utilised for the outlier check is straightforward. For each measurement, an arbitrary number of conditions may be specified. In a typical case, there will be a lower and an upper bound. While the utilised outlier detection method is simple, various more advanced methods exist as discussed in (Hodge and Austin, 2004), for example. An XML format has been designed to capture the outlier limits for each data item. The documents specifying the limits may be managed anywhere; fresh versions will be delivered to machines for utilisation. That is, even though a machine fleet may be geographically distributed, centralised outlier limit management is enabled.

To enable configurability, conditions for appearance items are defined with the same XML format as the outlier limits. For each appearance item, an arbitrary set of data items may be inspected. For each data item, there can be an arbitrary number of conditions (similar to each data item that may have multiple outlier conditions).

While various context recognition methods exist, the prototype utilises a simple though configurable way. The prevailing context is determined by finding the most typical stem data item collection category. That category is considered the context; any other data item collections are excluded from further processing as they are considered exceptions in the current environment. Categories are defined using a tree-like condition set (see Figure 5): the categorisation tree may inspect any data items to resolve the category of a data item collection. The categorisation tree is stored in a structured text document; it has been generated by a data analysis software in the enterprise office. The prototype parses the categorisation tree so it is available in the application during machine operation. Similar to outlier and appearance condition definitions, even the categorisation tree is delivered as a configuration file to each machine.

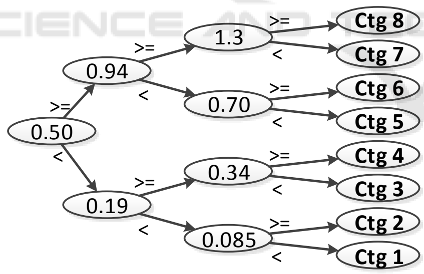

Figure 5: An example of categorising a tree stem after its volume in m^3 (though there could be multiple variables observed in the conditions). Here, the categories have indices from 1 to 8, a high index indicating a large stem. For instance, category 4 has the stems with a volume within range [0.34-0.50[.

The classes of the prototype are illustrated in Figure 6. In the prototype, an abstraction called *item condition* is essential: it defines a condition for a data item (such as a measurement). Item conditions are utilised for both outlier checking and specifying appearance items. Each item condition is a part of an *item condition definition* (as a value may have multiple boundaries), and each item condition definition is a part of an *item condition definition set* (such as the conditions of an appearance item). Item conditions are stored in an XML configuration file parsed by the *item condition XML reader* class. *Appearance resolver* class resolves which appearances are true for each data item collection. The conditions for data item collection categorisation are parsed by the *categorisation tree parser* class.

The prototype has been implemented with Java though any other platform could be used as well. As long as component interfaces (such as configuration formats) are as specified, even heterogeneous platforms are possible within an enterprise.

5.2 Practical Experiment

The prototype has been utilised in the refinement of real operational forestry data. It has been included in a workflow that estimates machine performance and suggests machine parameter tuning in case the parameters seem non-optimal; the same scenario has already been considered in (Kannisto et al., 2015). As the number of machine parameters may reach hundreds in a modern machine, their optimisation is too difficult for a typical operator. That is, such information refinement has considerable added value to the operating enterprise. The actual parameter optimisation application utilises the outcome of the data preprocessing introduced in this paper. The software has been executed on a desktop computer with a data retrieval interface identical to a physical forestry machine. As real operating data is utilised, the setup is almost identical as if the application were run in the field.

Parameter optimisation is not a simple task as it requires multiple factors to be considered. The operating context and the type of work being performed may affect both which parameter values result in a good performance and the actual performance values. Large amounts of historical data should be analysed to generate reference sets of performance values and optimal parameter values. As machines keep operating, data should be continuously collected to refresh parameter related knowledge; as knowledge updates are delivered to multiple machines, ease in management becomes beneficial. Knowledge generation actions require both extensive domain expertise and advanced data refinement methods so they should be performed by a dedicated group of skilled personnel. The knowledge may be managed by, for instance, machine manufacturer or fleet operator.

Here, the function under parameter optimisation is automatic tree stem positioning in a wood processing implement. Stems are positioned to be cut into logs. Such a case suits well for parameter optimisa-

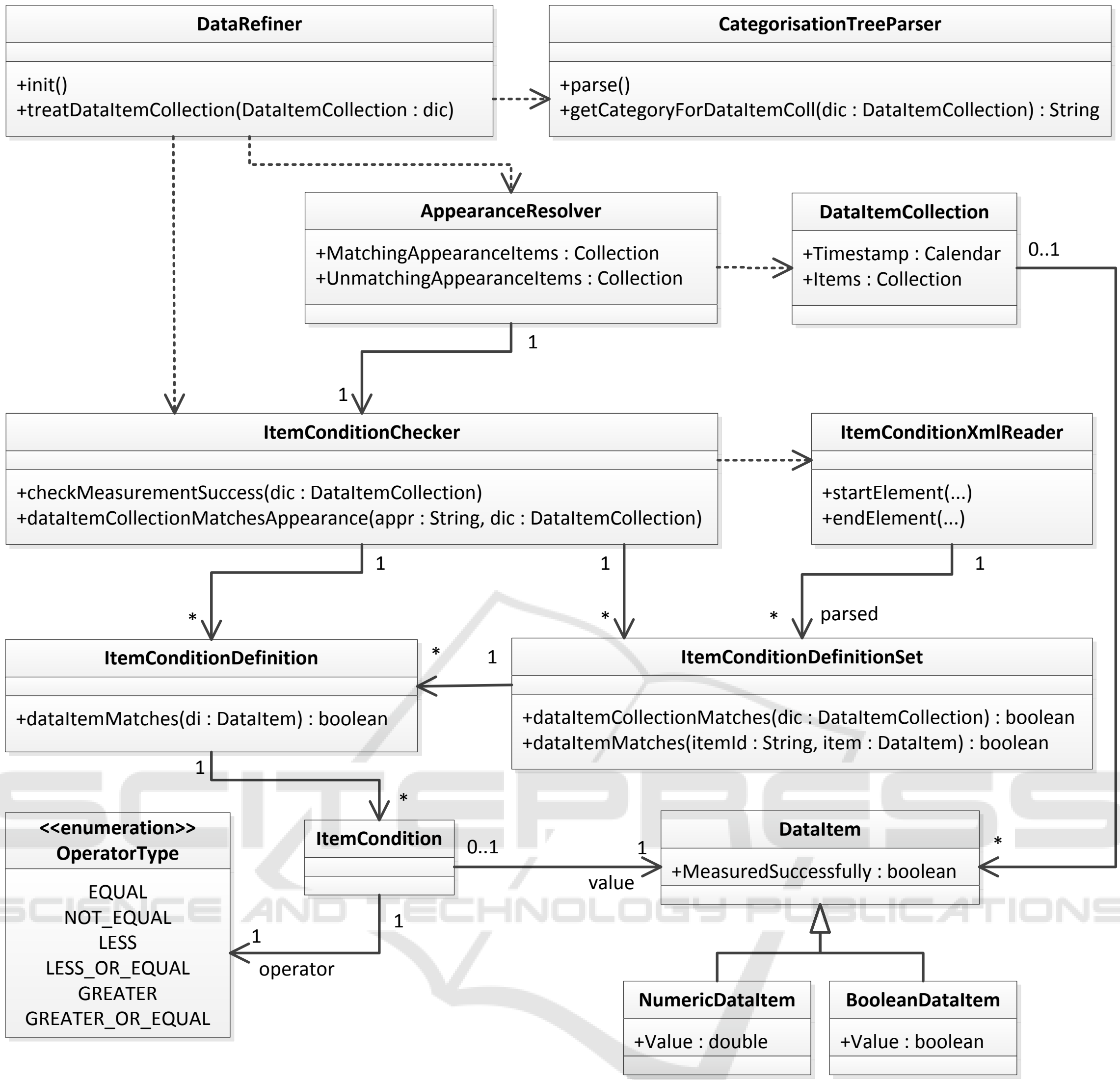

Figure 6: The classes of the prototype implementation.

tion as automatic positioning is controlled entirely by machine parameters rather than by the operator – the most of other machine functions are largely affected by operator skills.

The outliers of two measurements are observed in the experiment. *Positioning error* describes how close to its optimal cutting position a stem has been stopped. In contrast, *feed speed* does not determine positioning performance but it is an important measure as the overall machine performance is estimated in further data processing (more speed results in a higher productivity value). The outlier conditions are as follows: feed speed cannot be negative, and the absolute value of positioning error must be within 30 cm of the desired position.

Stem categorisation is important in the experiment. According to stem volume, each stem is put into one of eight categories. As little trees are not of interest in this felling scenario, there is an additional condition that each stem with a felling diameter of less than 15 cm is excluded. The context recognition method also uses the outcome of the categorisation. It is simplistic: for each category, there is a directly mapped context class. The stems in any other category are considered irrelevant and excluded from further processing.

In the experiment, appearance items have an informative function. They are generated using conditions that specify if a stem represents a long spruce or a long pine (that is, both tree species and stem length are observed). For the resulting Boolean *true* values, percentages are calculated how large their section is within the relevant stem category (or context; e.g. "64% of stems are long spruces"). While the parameter analysis application does not utilise these percentage values, the machine operator might want to observe himself if tree species or lengths actually affect optimal parameter values. If there are such factors, they should actually be discovered in fleet level data analyses. Then, they could be utilised by the parameter optimisation application in the field.

6 RESULTS AND DISCUSSION

The objective of this work was to design a software concept to enable centralised management of data refinement in an arbitrarily large geographically distributed machine fleet. Outlier inspection for measurements was required as well as data set categorisation and the possibility to specify variables derived from original data. Context recognition and consideration were also required.

The concept meets its requirements well. The ease of management of the application workflow was considered paramount: it is possible to configure not only outlier limits but also data set categorisation and the context determination method. As an enterprise may have machines in arbitrary geographic locations, the delivery of configuration data has also been considered. In addition, it is possible to specify variables for information that has been inferred from explicit measurement data. Such data may be numeric (calculated) or Boolean values resulting from the assertion of multiple conditions.

A functional data refinement component prototype has been implemented. It implements the specified data refinement flow. First, an outlier check is performed on measurement values followed by the calculation of derived variables. Then, each data item collection (a data set of key-value pairs) is categorised according to specified conditions, and finally, the prevailing context is determined using categorisation information. The configurability requirement is fulfilled by getting outlier conditions, derived variable calculation conditions and categorisation definition from externally defined configuration files.

The concept was experimented with real operative data from 11 forestry machines. For each machine, the data of thousands of stems was processed so it can be said there has been a lot of repetition in application cycles. The outcome of the software component (i.e. refined data) was utilised to optimise the parameters of automatic tree stem positioning in a wood processing implement. The data refinement results are in Table 1. In each data set, the number of stems in the context was relatively low. The context recognition method returned the same operating context for each data set (stems with volume within 0.19-0.34 m^3) so it is not included in the table.

The outlier results provided by the component seem useful. For positioning error values, the exclusion percentage is relatively low – mostly less than 1%, at most 1.4%. However, the highest exclusion percentage due to feed speed value is 9.7%. If these values were not excluded from further processing, they could cause significant errors in further calculations performed by other applications. Still, depending on error magnitudes, even a 1% section of erroneous values may cause misleading results.

18-54% of all stems were excluded from further processing as their felling diameter was less than 15 cm. The percentages are relatively high. As the parameter optimisation goal was concerned with the processing of large stems, such large amounts of relatively little stems might distort further calculations. However, it may also be asked if the processing of little stems should also be considered in optimisation. In that case, their data should be passed through distinguished from large stems.

The percentages of long spruces and pines are also included in the results table. In most cases, spruce appears the dominant species. The parameter analysis application did not utilise this information for anything so it is purely informative.

The context recognition method appeared to be ineffective as its result was the same context class for each test run. More context recognition and classification related research should be performed. The goal of context recognition should be reconsidered; that would specify which variables and what kind of methods should actually be included as the context is determined. However, the task is more related to domain expertise and data analysis rather than the knowledge management concept relevant in this study. In the end, it might be beneficial if the entire context recognition method could be updated along with the configuration.

The experiments made with the prototype indicate that the data refinement concept is functional. It has potential business value in real-life data processing: it would be easier to manage the refinement of the data consumed by various end user applications. Such applications may, for instance, assist in more effective machine operation. However, the prototype also has room for further development. It does not address the delivery of configuration documents – at least a loose framework might be beneficial. Still, the prototype provides a baseline for the delivery by having a spe-

Table 1: Data refinement results with real forestry machine operation data.

Mach ID	Stems	Logs	Feed speed outlier (logs)	Pos. error outlier (logs)	Stems excluded (felling diam <15 cm)	Stems in context	Long spruces (in context)	Long pines (in context)
1	11,000	27,000	4.0%	0.33%	54%	1,400	40%	52%
2	6,300	19,000	1.8%	1.1%	23%	1,200	60%	26%
3	14,000	39,000	4.1%	0.93%	36%	2,500	61%	22%
4	6,600	18,000	3.9%	0.56%	48%	1,100	61%	5.6%
5	5,900	18,000	2.9%	0.27%	31%	1,000	60%	8.7%
6	7,800	26,000	5.1%	0.36%	30%	1,100	75%	9.1%
7	8,000	27,000	1.6%	0.39%	26%	1,400	72%	7.9%
8	10,000	28,000	4.9%	0.76%	32%	2,000	33%	33%
9	12,000	38,000	4.9%	1.4%	34%	1,600	64%	20%
10	6,800	25,000	9.7%	0.93%	18%	1,100	55%	4.2%
11	6,500	20,000	4.9%	1.0%	29%	1,400	62%	13%

cific XML format for some configuration items. Also, derived variables can only be Boolean values – numeric values are not currently supported though they would offer significantly more potential for various uses cases.

7 CONCLUSION

This paper introduces a software concept for centrally manageable data refinement run locally in the machines of an arbitrarily large fleet. As machine data is utilised locally in end-user applications (such as feedback generation to improve machine operation and productivity), it is beneficial if the required data preprocessing is configurable and managed on the fleet level. Configurability covers multiple actions: outlier checks detect erroneous sensor output, derived variables can be calculated from original data, and data sets are categorised according to predefined conditions. Further, determining the operating context is also configurable. Context awareness is utilised as the context may affect how data should be interpreted.

A functional prototype has been implemented. Utilising externally defined configurations, it processes operational data retrieved from an interface similar to a physical production machine. The solution showed its potential as a part of an added-value data refinement concept by enabling centralised management.

As the current prototype does not cover all concept aspects, a few future tasks remain. A concrete solution for the delivery of configuration data from office to machines should be designed. Also, the current context recognition method appeared to be too simple.

REFERENCES

Banerjee, T. P. and Das, S. (2012). Multi-sensor data fusion using support vector machine for motor fault detection. *Information Sciences*, 217:96 – 107.

Basir, O. and Yuan, X. (2007). Engine fault diagnosis based on multi-sensor information fusion using dempster-shafer evidence theory. *Information Fusion*, 8(4):379 – 386.

Choudhury, T., Consolvo, S., Harrison, B., Hightower, J., Lamarca, A., Legrand, L., Rahimi, A., Rea, A., Bordello, G., Hemingway, B., Klasnja, P., Koscher, K., Landay, J., Lester, J., Wyatt, D., and Haehnel, D. (2008). The mobile sensing platform: An embedded activity recognition system. *Pervasive Computing, IEEE*, 7(2):32–41.

Duan, L. and Xu, L. D. (2012). Business intelligence for enterprise systems: A survey. *Industrial Informatics, IEEE Transactions on*, 8(3):679–687.

Favela, J., Tentori, M., Castro, L. A., Gonzalez, V. M., Moran, E. B., and Martínez-García, A. I. (2007). Activity recognition for context-aware hospital applications: Issues and opportunities for the deployment of pervasive networks. *Mob. Netw. Appl.*, 12(2-3):155–171.

Fountas, S., Sorensen, C., Tsiropoulos, Z., Cavalaris, C., Liakos, V., and Gemtos, T. (2015). Farm machinery management information system. *Computers and Electronics in Agriculture*, 110:131–138.

Golparvar-Fard, M., Heydarian, A., and Niebles, J. C. (2013). Vision-based action recognition of earthmoving equipment using spatio-temporal features and support vector machine classifiers. *Advanced Engineering Informatics*, 27(4):652 – 663.

Hodge, V. and Austin, J. (2004). A survey of outlier detection methodologies. *Artificial Intelligence Review*, 22(2):85–126.

Hou, L. and Bergmann, N. (2012). Novel industrial wireless sensor networks for machine condition monitoring and fault diagnosis. *Instrumentation and Measurement, IEEE Transactions on*, 61(10):2787–2798.

Iftikhar, N. and Pedersen, T. B. (2011). Flexible exchange of farming device data. *Computers and Electronics in Agriculture*, 75(1):52 – 63.

Jardine, A. K., Lin, D., and Banjevic, D. (2006). A review on machinery diagnostics and prognostics implementing condition-based maintenance. *Mechanical Systems and Signal Processing*, 20(7):1483 – 1510.

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.

Kannisto, P., Hästbacka, D., Vilkko, M., and Kuikka, S. (2015). Service architecture and interface design for mobile machine parameter optimization system. *IFAC-PapersOnLine*, 48(3):848 – 854. 15th IFAC Symposium on Information Control Problems in Manufacturing INCOM 2015.

Khot, L. R., Tang, L., Blackmore, S., and Nørremark, M. (2006). Navigational context recognition for an autonomous robot in a simulated tree plantation. *Transactions of the ASABE*, 49(5):1579–1588.

LaValle, S., Lesser, E., Shockley, R., Hopkins, M. S., and Kruschwitz, N. (2011). Big data, analytics and the path from insights to value. *MIT sloan management review*, 52(2):21–31.

Lu, B. and Gungor, V. (2009). Online and remote motor energy monitoring and fault diagnostics using wireless sensor networks. *Industrial Electronics, IEEE Transactions on*, 56(11):4651–4659.

Osborne, J. W. and Overbay, A. (2004). The power of outliers (and why researchers should always check for them). *Practical assessment, research & evaluation*, 9(6).

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

Peets, S., Mouazen, A. M., Blackburn, K., Kuang, B., and Wiebensohn, J. (2012). Methods and procedures for automatic collection and management of data acquired from on-the-go sensors with application to on-the-go soil sensors. *Computers and Electronics in Agriculture*, 81:104 – 112.

Steinberger, G., Rothmund, M., and Auernhammer, H. (2009). Mobile farm equipment as a data source in an agricultural service architecture. *Computers and Electronics in Agriculture*, 65(2):238 – 246.

Stiefmeier, T., Roggen, D., Ogris, G., Lukowicz, P., and Tröster, G. (2008). Wearable activity tracking in car manufacturing. *IEEE Pervasive Computing*, 7(2):42–50.

Väyrynen, T., Peltokangas, S., Anttila, E., and Vilkko, M. (2015). Data-driven approach for analysis of performance indices in mobile work machines. In *DATA ANALYTICS 2015, The Fourth International Conference on Data Analytics*, pages 81–86.

Yang, B.-S. and Kim, K. J. (2006). Application of dempstershafer theory in fault diagnosis of induction motors using vibration and current signals. *Mechanical Systems and Signal Processing*, 20(2):403 – 420.