Kannisto, P. & Hästbacka, D., "Cloud-based Management of Machine Learning Generated Knowledge for Fleet Data Refinement", in Fred, A., Dietz, J., Aveiro, D., Liu, K., Bernardino, J. & Filipe, J., Eds., Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2016; Communications in Computer and Information Science, vol 914), 2019, pp. 267-286. DOI: 10.1007/978-3-319-99701-8_13

Cloud-based Management of Machine Learning Generated Knowledge for Fleet Data Refinement

Petri Kannisto and David Hästbacka

Tampere University of Technology, P.O. Box 692, FI-33101 Tampere, Finland,
(petri.kannisto|david.hastbacka)@tut.fi,
WWW home page: http://www.tut.fi

Abstract. The modern mobile machinery has advanced on-board computer systems. They may execute various types of applications observing machine operation based on sensor data (such as feedback generators for more efficient operation). Measurement data utilisation requires preprocessing before use (e.g. outlier detection or dataset categorisation). As more and more data is collected from machine operation, better data preprocessing knowledge may be generated with data analyses. To enable the repeated deployment of that knowledge to machines in operation, information management must be considered; this is particularly challenging in geographically distributed fleets. This study considers both data refinement management and the refinement workflow required for data utilisation. The role of machine learning in data refinement knowledge generation is also considered. A functional cloud-managed data refinement component prototype has been implemented, and an experiment has been made with forestry data. The results indicate that the concept has considerable business potential.

Keywords: Distributed Knowledge Management, Mobile Machinery, Cloud Services, Data Preprocessing, Machine Learning

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. [19]) but also in industrial context (Duan & Xu [6]). 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 machinery, machine learning can be applied for multiple use cases. It may not only generate added value from data but it may also aid the generation of data *preprocessing* knowledge that serves other data analysis tasks.

In this paper, a software concept is introduced – intended for service architectures – to enable the centralised management of fleet-wide 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.

This work has two main contributions: a conceptual cloud service architecture with machine learning and a functional prototype. The conceptual architecture utilises cloud services for storing extensive amounts of data as well as for machine learning to generate novel knowledge. The prototype covers an intermediary component that refines measurement data locally in machines – it receives its configuration from access points in a cloud so centralised management is achieved. The component accomplishes essential first-hand tasks thus generating information and facilitating further data analysis in end user applications.

The utilised research method is design science research. Novel knowledge is generated by designing artefacts that are evaluated against their requirements (referred to by e.g. March & Smith [22]).

This article is a revised version of a conference paper already published by the authors [15]. The original concept has been extended with cloud services and machine learning aspects, and some of the original contributions have also been more comprehensively explained.

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

2 Related Work

Among the publications in the industrial domain, no work has been found with a similarly extensive combination of a data processing workflow, cloud-based configurability and a data analysis or machine learning aspect. Various studies have been published with some common aspects though; thus, this part summarises the work related to either cloud services in production systems, machine data refinement, equipment data exchange or context awareness.

The Vehicular Cloud Computing (VCC) concept combines distributed data processing and mobility with a point of view different to this work. Its idea is to utilise the on-board computation and sensing capabilities of vehicles to

enhance, for instance, traffic safety and management. Whaiduzzaman et al. [31] have written an extensive survey about the topic.

Storing machine or vehicle data in cloud is also a resource for large-scale data analysis. Bahga & Madisetti [1] have studied storing industrial measurement data in cloud to run analyses to raise maintenance performance. Filev et al. [8] show how vehicular data may be collected to a cloud and assisting services may be provided back to vehicles.

Even though both cloud and industrial production are related to this work, the Cloud Manufacturing concept is more related to business collaboration and interoperability within manufacturing networks. In manufacturing, the cloud service paradigm is expected to bring benefits such as scalability, agility and easier business networking. Tao et al. [28] have primarily envisioned manufacturing resource services while Wu et al. [32] have also covered product design as a cloud service.

Farming equipment related data collection or exchange has been researched in various papers. In a study by Steinberger et al. [26], farming equipment data is exposed in a service architecture. A work by Iftikhar & Pedersen [13] includes device data exchange in a bidirectional manner between office computers and farming machines. Peets et al. [25] provide a solution for data collection from various types of sensors. Fountas et al. [9] have introduced an information system concept for the management of farming machines. Machine data retrieval and integration concerns are present even in this work.

There are also other publications related to mobile machinery data processing. Palmroth [24] has studied the analysis of mobile machine data to assist operator learning. A knowledge management solution for operator performance assessment in the field is considered by Kannisto et al. [17]. Kannisto et al. [16] have introduced a system architecture 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, and cloud services have not been considered in the implementations.

Fault diagnostics and condition monitoring methods are related as they consider information generation by processing measured data. Various mathematical methods can be utilised for diagnostics as presented by Banerjee & Das, Basir & Yuan and Yang & Kim [2, 3, 33]. Condition-based maintenance (CBM) is enabled by utilising collected condition data as proposed by Jardine et al. [14]. Recently, even wireless sensor networks (WSN) have been utilised in diagnostics as suggested by Hou & Bergmann and Lu & Gungor [12, 21]. These studies focus on data processing methods 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. Khot et al. [18] provide 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 the work of Golparvar-Fard et al. [10] where earthmoving equipment actions are recognised from video. Human activities recognition has also been researched including hospital work (Favela et al. [7]), car manufacturing (Stiefmeier et al. [27]) and general activities (Choudbury et al. [4]). Wan et al. [30] have even considered vehicular context recognition applications for parking assistance, vehicle routing and hazard prediction. 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

3.1 Opportunities and Challenges of Data Analysis and Machine Learning

In the pursue for more efficient machine operation, this study recognises two data analysis use cases: fleet-wide and machine specific. The fleet-wide use case considers what is common for an entire group of machines. By utilising appropriate data analysis methods, multiple machine data sets may be processed together even if there were significant differences in machine types, operating environments and work types. In contrast, machine specific data analysis aims at discovering how a particular machine differs from the rest. Such differences appear due to the variation of machine parts: for instance, hydraulic components may vary even if they represented the same product, and a machine may have encountered more equipment wear than most similar machines.

To have a restricted scope, this work is concerned with the *information management* of fleet-wide data analysis for data preprocessing purposes. That is, while important, data analysis in individual machines, the actual data analysis methods as well as any end user applications are not in the scope (see Table 1). Still, end user applications bring the ultimate benefit to operators: the applications build added value by providing – for instance – assistance for machine operation.

Table 1. The scope of the work within machinery data utilisation.

Data analysis aspect	***Information management aspect***
Fleet-wide scope	Single machine scope
End user applications	***Data preprocessing***

While various data analysis methods could be utilised, this study emphasises the possibilities of machine learning. As huge amounts of operational machine fleet data are collected, machine learning methods may reveal new knowledge that might otherwise remain unobserved. That is due to the incomplete and ever-evolving nature of domain expertise – not only knowledge coverage improves but

also new advances in machinery technology cause repeated changes. Especially, *deep learning* should be applied: it enables effective machine learning by utilising multiple abstraction levels (as stated by Deng & Yu [5, pp. 205-206] and LeCun et al. [20]).

Whichever are the end-user applications that utilise machine data, fleet-wide utilisation sets multiple requirements to data preprocessing and its management. 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 fleet (see Figure 1). Data collection enables analysis using fleet-wide data, producing data refinement configurations to be utilised locally in individual machines. The configurations are then utilised in data preprocessing for end user applications. Modern mobile machinery have advanced information systems and multiple sensors installed so a large amount of measurement data is produced during a work cycle, not to mention an entire work shift or weeks of operation. The more machines there are, the more data is generated. How distributed are the machines actually – may they operate anywhere in the world? What if the machines have no persistent internet connectivity?

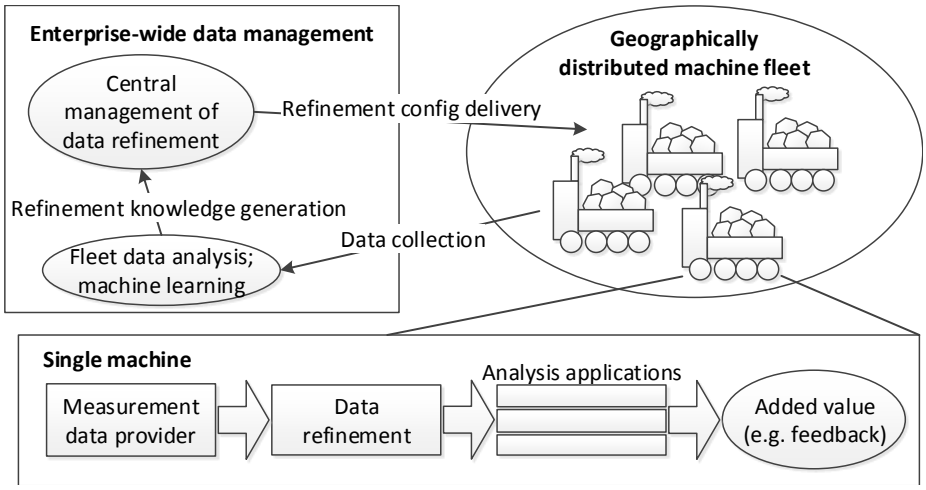

Fig. 1. Data collection, analysis using fleet-wide data, refinement configuration management and data analysis results utilisation locally in machines. Adapted from [15].

This work is particularly motivated by forestry. Tree stem processing is a demanding task in terms of optimisation; there may be a lot of variation between forests and terrains even inside a geographically restricted area; also, there is often no internet connectivity in forests. Thus, the requirements of the next subsection give various forestry related examples.

3.2 Preprocessing Management Requirements

This study aims at data *preprocessing* management as it is typically required for sensor data. The scope is more extensive than just individual measurements as various more advanced features are beneficial for end user applications.

Data is structured as data sets called *data item collections*. A data item collection contains all the measurement values saved at a certain point in time. In addition, as modern machines have multiple operation related parameters (often customisable by the operator: such as the maximum power supplied to actuators), they are also stored. Thus, a data item collection provides 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 the prevailing operating context.

In the forestry example, a data item collection represents the data of a single tree stem. For each stem, modern equipment supply various measurements such as felling diameter, stem length or how quickly the stem has been processed with the machine. The measurements and all machine parameters will be stored in a data item collection so stem data sets may be processed easily, one by one.

Machine type specific data item collection processing is likely required. First, variation is expected between machine types in measurement availability. 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 and its implements. Parameter sets may vary as well as how a certain parameter affects machine operation.

Measurement failures must also be considered. 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 has been discussed by, for instance, Osborne & Overbay [23].

Some variables cannot be measured as such but they have to be calculated. For instance, even if there were a measurement value for the productivity during a single work cycle, the daily productivity must be summed over a day. Further, a machine may change its position multiple times during a day, and working conditions may be so dirty that the windscreen must be cleaned multiple times during a work shift. If a productivity variable should only cover the actual material processing, any idle periods are to be excluded from productivity consideration. In forestry, an indirectly calculated Boolean flag may be utilised to inspect the tree species and size to help limiting data processing to a particular tree category.

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. However, if it is possible to submit data often to cloud, space is typically not a problem.

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 – the context may significantly affect how a machine can perform as argued by Väyrynen et al. [29]. 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.

Due to machine fleet distribution, data caching is important. First, the requirement applies to configuration delivery: the data refinement application cannot rely on network connectivity so it needs local copies for any configuration items. Second, as measurement data is collected from machines for future data analysis activity, similar caching is required so the data can wait for delivery to the enterprise cloud.

The various requirements and related specification items are summarised in Table 2. The required data preprocessing tasks cover e.g. data structures, indirect measure calculation and contextual variation.

Table 2. Data preprocessing requirements summarised.

Requirement	Conforming specification item
Associate related data in sets	Use data item collections
Machine types have differences	Consider machine types in data processing
Measurement errors reduce analysis reliability	Data outlier analysis
Indirectly calculated measures	Support for derived variables
Allow data calculation evolution for old data	Store raw measurement values as such
Distinction and grouping of data sets	Data item collection categorisation
Operating environment and work type differences	Context recognition support
No persistent internet connectivity	Data caching

4 Managing Data Refinement with Cloud Services and Machine Learning

4.1 Refinement 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.

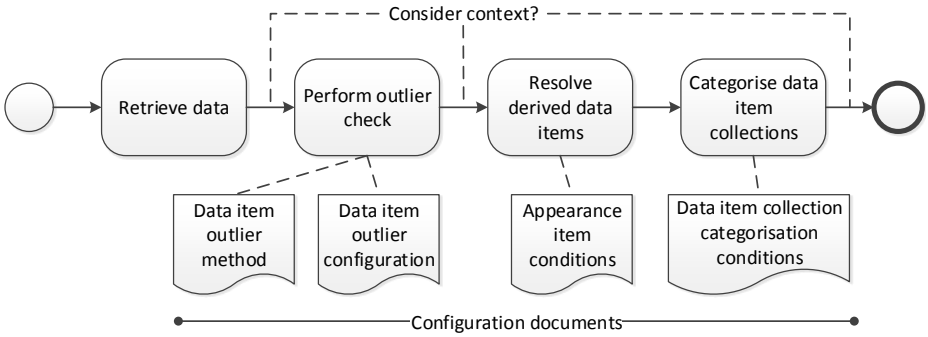

Fig. 2. Data refinement flow. Adapted from [15].

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.

The next phase covers the calculation of derived variables (i.e. the data not directly measurable). 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 the species of a stem is spruce. The information may be utilised in further data analysis to easily determine which stems are interesting – for example, occasional birches in a spruce forest may be ignored.

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 certain phases, its design is adaptable in terms of context awareness.

Let us consider forestry again to have a workflow execution example. 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 design choice depends on the application.

4.2 Cloud-Based Configuration Management with Machine Learning

Data refinement configuration management is illustrated in Figure 3. The number of machines is arbitrary as well as their geographic locations. Various appli-

cations may utilise refined machine data, but the aspects of managing the actual refinement are explained in the following paragraphs.

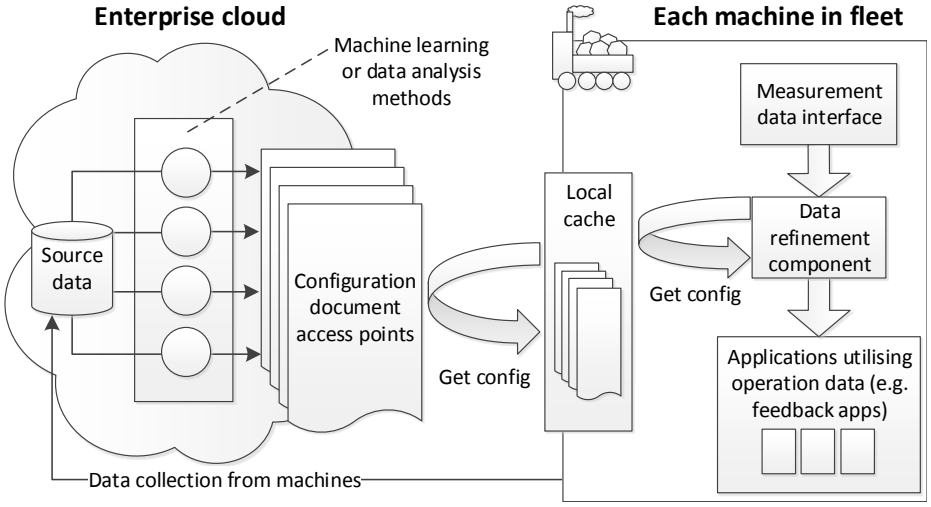

Fig. 3. Data analysis and data refinement management illustrated. Adapted from [15].

A software component has been designed to implement the data refinement workflow that utilises externally provided configuration documents. In each individual machine, it retrieves raw measurement data from the measurement data interface of the machine. Due to internet connection limitations, a cache holds local copies of the prevailing refinement configuration retrieved from the enterprise cloud. Having a software *component* enables reuse for the functionality in an arbitrary number of applications.

The enterprise cloud has multiple tasks in the data refinement management concept. First, it maintains a centralised storage for machine data. A large coverage is required for effective fleet-wide information generation. Second, utilising the stored data, machine learning or other data analysis methods are applied to generate the data refinement configuration utilised locally in machines. Multiple analysis methods are required as there are various configuration items. Third, the cloud stores the analysis results – i.e. the refinement configuration documents – and provides access points to make them available for machines. The everyday technology portfolio covers various networking methods for configuration retrieval such as HTTP (Hypertext Transfer Protocol) widely supported by software libraries. In the end, the cloud paradigm provides a basis for centralised management and scalable business in an environment where the data amount is huge and distribution requirements are ultimate.

5 Cloud-Enabled Data Refinement Prototype

5.1 Concrete Data Refinement

Following the specified concept, a prototype has been implemented for tree stem data processing in the forestry domain (the data refinement flow 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.

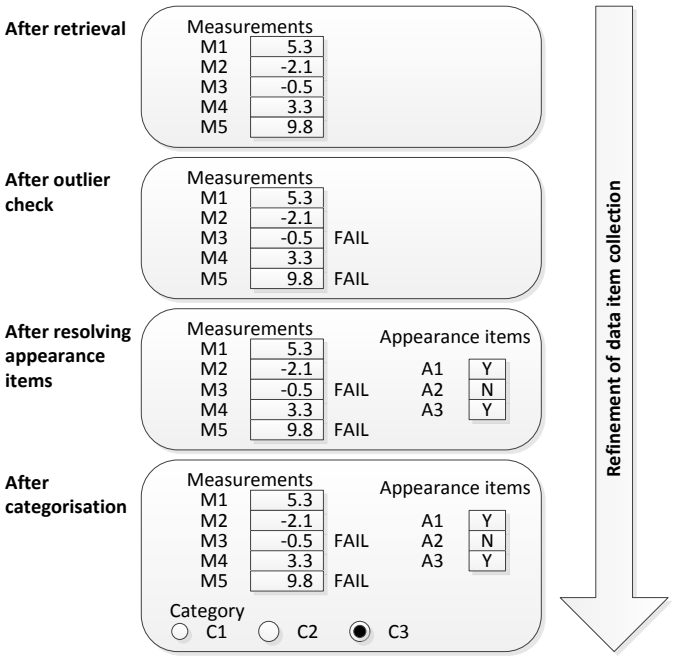

Fig. 4. Data refinement in the prototype implementation [15].

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 by Hodge & Austin [11], for example. An XML (Extensible Markup Language) format has been designed to have configurable outlier conditions for each data item.

To enable configurability, the 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 generated in a fleet-wide data analysis. 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 transferred as a configuration item to each machine.

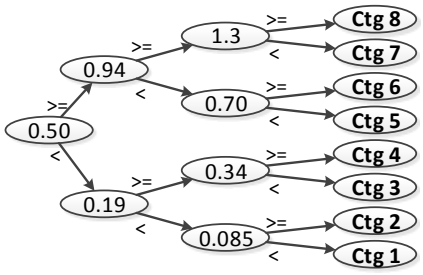

Fig. 5. An example of categorising a tree stem after its volume in m^3 (though there could be multiple variables observed in the conditions). Here, the categories have indices from 1 to 8, a high index indicating a large stem. For instance, category 4 has the stems with a volume within range [0.34-0.50[. [15]

5.2 Software Implementation

Figure 6 illustrates the concrete software implementation of the prototype. The prototype may be roughly divided to a cloud side and a machine side; both the sides are explained in more detail in the following paragraphs.

The cloud side covers machine learning functions as well as data refinement configuration access points. The utilised cloud environment is Microsoft Azure.

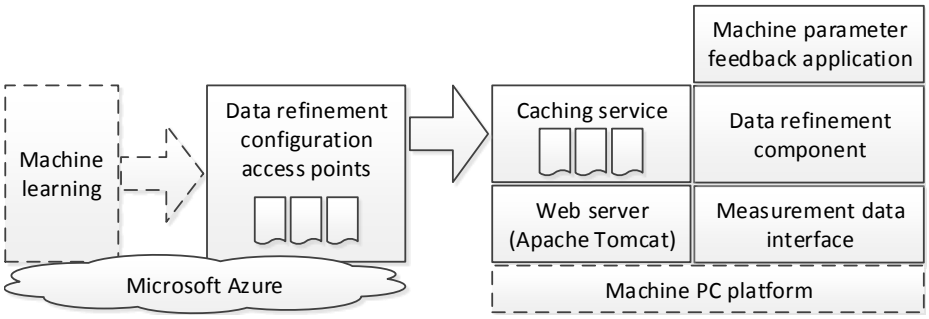

Fig. 6. Prototype architecture.

In the prototype, no machine learning is performed in the cloud as it is out of the scope of this study. Still, as Azure has the facilities to store large amounts of data and even machine learning capabilities, it is considered an appropriate platform. All the configuration items (the conditions for measurement outliers, appearance items and tree stem categorisation) are located in Azure to demonstrate their accessibility from a HTTP-based REST API in the cloud.

Due to non-persistent internet connectivity, a caching web service has been implemented to provide an access to configuration items locally in machinery. The web service has been implemented with Java and it is run on a Tomcat web server in a desktop computer. Modern machinery often run their equipment and operation related software on a PC platform, which makes it possible to install a general-purpose web server even there.

The actual configurable data refinement component has been utilised in an application that assists the machine operator to optimise various equipment parameters during machine operation. Although run in a desktop PC, the execution environment is realistic as a measurement data interface identical to a physical machine is utilised; besides, the interface has been set up to provide data collected during actual physical machine operation.

The classes of the data refinement component prototype are illustrated in Figure 7. 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 data refinement component has been implemented with Java although any other platform could be used as well. As long as component interfaces (such

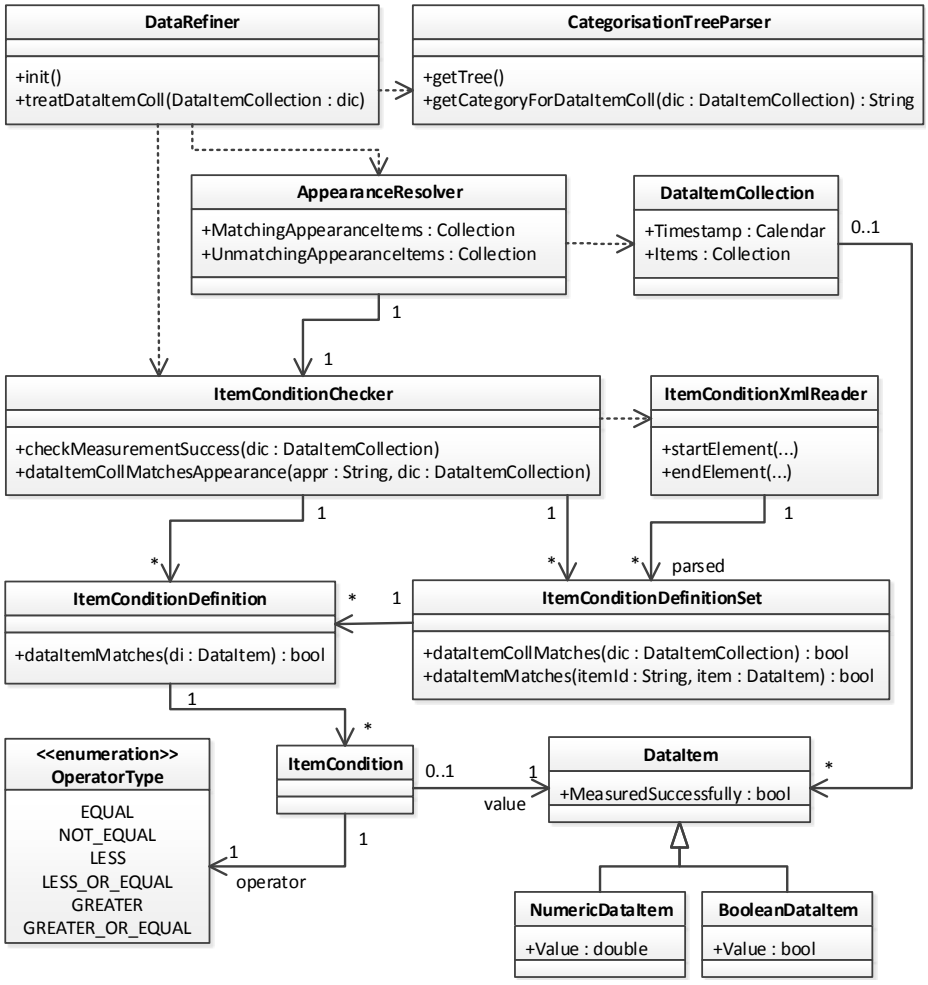

Fig. 7. The classes of the data refinement component prototype. Adapted from [15].

as configuration formats) are as specified, even heterogeneous platforms are possible within an enterprise.

5.3 Practical Experiment with Machine Data

The prototype has been utilised in the refinement of real operational forestry data in a machine parameter optimisation application. The application estimates machine performance and suggests parameter tuning in case the parameters seem non-optimal. As the number of machine parameters may reach hundreds in a modern machine, their optimisation is 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. As real operating data and realistic interfaces are utilised, the setup is almost identical as if the application were run in the field. Kannisto et al. [16] have already considered the scenario with parameters rather than data preprocessing in the scope.

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, the machine manufacturer or a fleet operator.

In this demonstration, 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 optimisation 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 themselves 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. From the conceptual point of view, the configurable indirect variable calculation feature improves management possibilities in data preprocessing.

6 Results and Discussion

The objective of this work was to design a software concept to enable the 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 calculated from original data. Context recognition and consideration were also required.

The concept meets its information management 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 recognition method. In addition, it is possible to specify variables for information inferred from explicit measurement data. Such data may be numeric (calculated) or Boolean values (resulting from the assertion of multiple conditions). The concept enables data collection from machines, machine learning to generate the configuration items as well as access to the configuration items managed in a cloud environment.

A functional cloud-managed data refinement software 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 a cloud service. Machine mobility and geographic distribution have also been considered by implementing a caching service run locally in each machine.

The concept has been experimented with real operative data from 11 forestry machines. For each machine, the data of thousands of stems was processed so 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 3. 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.

Table 3. Data refinement results with real forestry machine operation data [15].

Mach ID	Stems	Logs	Feed sp. outlier (logs)	Pos. error outlier (logs)	Stems excluded (felling diam <15 cm)	Stems in context	Long spruces (context)	Long pines (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%	27%	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%

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 in the experiment.

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 management concept is functional and valuable. 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. Context recognition should be studied further to provide more practical value. Derived variables can only be Boolean values – numeric values are not currently supported though they would offer significantly more potential for various uses cases. In addition, even though configuration documents are already managed with cloud services, their coupling with concrete machine learning methods in the cloud are not covered. The prototype should be developed further to cover the entire chain of data collection, data storage and machine learning chain. While data analysis may be applied in any environment, a cloud promotes scalability and availability, which is beneficial for a large enterprises in a distributed environment. Ultimately, it would be interesting to see the concept in operation in an everyday business environment. Finally, a long-term development need is the consideration of individual machine characteristics. In practice, individual differences may affect machine performance and sensor readings – this stems from, for instance, individual hydraulic component characteristics or the degree of abrasion. This also affects how raw sensor data should be preprocessed. In the future, machine learning should be applied *locally* in each machine to consider such differences.

7 Conclusion

In this study, a software system concept is introduced to enable centralised management for measurement data refinement within a distributed machine fleet. Modern machines have been equipped with advanced ICT devices that enable added-value software for various purposes (such as operator feedback for more efficient operation). To ease application development, the data refinement concept covers configurability for multiple important data preprocessing tasks including outlier detection, the calculation of derived variables and context recognition. From the management point of view, the concept covers measurement data collection, the utilisation of machine learning methods to generate data refinement configurations as well as configuration item access points – all in a cloud.

Following the concept, a functional data refinement management prototype has been implemented. It is an intermediary component that refines measurement data using configuration items received from cloud services. The prototype has been executed as a part of an application that provides assistance in

machine parametrisation. Experiments with real operational measurement data have demonstrated the practical value of the concept: how machine data refinement management can be largely facilitated with cloud services.

There are also future research tasks. While successful, the prototype should be developed further to meet all the requirements of the concept. Also, the concept should cover even machine learning run locally in machines to consider individual machine characteristics.

Acknowledgments. This work was made as a part of the D2I (Data to Intelligence) project funded by Tekes (the Finnish Funding Agency for Innovation). The authors would like to express their sincere gratitude to the project partners and participant companies.

References

1. Bahga, A., Madisetti, V.K.: Analyzing massive machine maintenance data in a computing cloud. IEEE Transactions on Parallel and Distributed Systems **23**(10), 1831–1843 (2012). DOI 10.1109/TPDS.2011.306
2. Banerjee, T.P., Das, S.: Multi-sensor data fusion using support vector machine for motor fault detection. Information Sciences **217**, 96 – 107 (2012). DOI http://dx.doi.org/10.1016/j.ins.2012.06.016
3. Basir, O., Yuan, X.: Engine fault diagnosis based on multi-sensor information fusion using Dempster-Shafer evidence theory. Information Fusion **8**(4), 379 – 386 (2007). DOI http://dx.doi.org/10.1016/j.inffus.2005.07.003
4. 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., Haehnel, D.: The mobile sensing platform: An embedded activity recognition system. Pervasive Computing, IEEE **7**(2), 32–41 (2008). DOI 10.1109/MPRV.2008.39
5. Deng, L., Yu, D.: Deep learning: Methods and applications. Foundations and Trends in Signal Processing **7**(34), 197–387 (2014). DOI 10.1561/2000000039
6. Duan, L., Xu, L.D.: Business intelligence for enterprise systems: A survey. Industrial Informatics, IEEE Transactions on **8**(3), 679–687 (2012). DOI 10.1109/TII.2012.2188804
7. Favela, J., Tentori, M., Castro, L.A., Gonzalez, V.M., Moran, E.B., Martínez-García, A.I.: Activity recognition for context-aware hospital applications: Issues and opportunities for the deployment of pervasive networks. Mob. Netw. Appl. **12**(2-3), 155–171 (2007). DOI 10.1007/s11036-007-0013-5
8. Filev, D., Lu, J., Hrovat, D.: Future mobility: Integrating vehicle control with cloud computing. Mechanical Engineering **135**(3), S18–S24 (2013)
9. Fountas, S., Sorensen, C., Tsiropoulos, Z., Cavalaris, C., Liakos, V., Gemtos, T.: Farm machinery management information system. Computers and Electronics in Agriculture **110**, 131–138 (2015). DOI http://dx.doi.org/10.1016/j.compag.2014.11.011
10. Golparvar-Fard, M., Heydarian, A., Niebles, J.C.: Vision-based action recognition of earthmoving equipment using spatio-temporal features and support vector machine classifiers. Advanced Engineering Informatics **27**(4), 652 – 663 (2013). DOI http://dx.doi.org/10.1016/j.aei.2013.09.001

11. Hodge, V., Austin, J.: A survey of outlier detection methodologies. Artificial Intelligence Review **22**(2), 85–126 (2004). DOI 10.1007/s10462-004-4304-y
12. Hou, L., Bergmann, N.: Novel industrial wireless sensor networks for machine condition monitoring and fault diagnosis. Instrumentation and Measurement, IEEE Transactions on **61**(10), 2787–2798 (2012). DOI 10.1109/TIM.2012.2200817
13. Iftikhar, N., Pedersen, T.B.: Flexible exchange of farming device data. Computers and Electronics in Agriculture **75**(1), 52 – 63 (2011). DOI http://dx.doi.org/10.1016/j.compag.2010.09.010
14. Jardine, A.K., Lin, D., Banjevic, D.: A review on machinery diagnostics and prognostics implementing condition-based maintenance. Mechanical Systems and Signal Processing **20**(7), 1483 – 1510 (2006). DOI http://dx.doi.org/10.1016/j.ymssp.2005.09.012
15. Kannisto, P., Hästbacka, D.: Enabling centralised management of local sensor data refinement in machine fleets. In: Proceedings of the 8th International Conference on Knowledge Management and Information Sharing, vol. 3, pp. 21–30 (2016)
16. Kannisto, P., Hästbacka, D., Kuikka, S.: System architecture for mastering machine parameter optimisation. Computers in Industry **85**, 39 – 47 (2017). DOI http://dx.doi.org/10.1016/j.compind.2016.12.006
17. Kannisto, P., Hästbacka, D., Palmroth, L., Kuikka, S.: 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, pp. 440–449 (2014). DOI 10.5220/0004870004400449
18. Khot, L.R., Tang, L., Blackmore, S., Nørremark, M.: Navigational context recognition for an autonomous robot in a simulated tree plantation. Transactions of the ASABE **49**(5), 1579–1588 (2006)
19. LaValle, S., Lesser, E., Shockley, R., Hopkins, M.S., Kruschwitz, N.: Big data, analytics and the path from insights to value. MIT sloan management review **52**(2), 21–31 (2011)
20. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature **521**(7553), 436–444 (2015). DOI doi:10.1038/nature14539
21. Lu, B., Gungor, V.: Online and remote motor energy monitoring and fault diagnostics using wireless sensor networks. Industrial Electronics, IEEE Transactions on **56**(11), 4651–4659 (2009). DOI 10.1109/TIE.2009.2028349
22. March, S.T., Smith, G.F.: Design and natural science research on information technology. Decision Support Systems **15**(4), 251 – 266 (1995). DOI http://dx.doi.org/10.1016/0167-9236(94)00041-2
23. Osborne, J.W., Overbay, A.: The power of outliers (and why researchers should always check for them). Practical assessment, research & evaluation **9**(6) (2004)
24. Palmroth, L.: Performance monitoring and operator assistance systems in mobile machines. Ph.D. thesis, Department of Automation Science and Engineering, Tampere University of Technology, Tampere, Finland (2011)
25. Peets, S., Mouazen, A.M., Blackburn, K., Kuang, B., Wiebensohn, J.: 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 (2012). DOI http://dx.doi.org/10.1016/j.compag.2011.11.011
26. Steinberger, G., Rothmund, M., Auernhammer, H.: Mobile farm equipment as a data source in an agricultural service architecture. Computers and Electronics in Agriculture **65**(2), 238 – 246 (2009). DOI http://dx.doi.org/10.1016/j.compag.2008.10.005

27. Stiefmeier, T., Roggen, D., Ogris, G., Lukowicz, P., Tröster, G.: Wearable activity tracking in car manufacturing. IEEE Pervasive Computing **7**(2), 42–50 (2008). DOI http://doi.ieeecomputersociety.org/10.1109/MPRV.2008.40
28. Tao, F., Zhang, L., Liu, Y., Cheng, Y., Wang, L., Xu, X.: Manufacturing service management in cloud manufacturing: Overview and future research directions. Journal of Manufacturing Science and Engineering **137**(4) (2015). DOI 10.1115/1.4030510
29. Väyrynen, T., Peltokangas, S., Anttila, E., Vilkko, M.: Data-driven approach for analysis of performance indices in mobile work machines. In: Data Analytics 2015, The Fourth International Conference on Data Analytics, pp. 81–86 (2015)
30. Wan, J., Zhang, D., Zhao, S., Yang, L.T., Lloret, J.: Context-aware vehicular cyber-physical systems with cloud support: architecture, challenges, and solutions. IEEE Communications Magazine **52**(8), 106–113 (2014). DOI 10.1109/MCOM.2014.6871677
31. Whaiduzzaman, M., Sookhak, M., Gani, A., Buyya, R.: A survey on vehicular cloud computing. Journal of Network and Computer Applications **40**, 325 – 344 (2014). DOI http://dx.doi.org/10.1016/j.jnca.2013.08.004
32. Wu, D., Rosen, D.W., Wang, L., Schaefer, D.: Cloud-based design and manufacturing: A new paradigm in digital manufacturing and design innovation. Computer-Aided Design **59**, 1 – 14 (2015). DOI http://dx.doi.org/10.1016/j.cad.2014.07.006
33. Yang, B.S., Kim, K.J.: Application of Dempster-Shafer theory in fault diagnosis of induction motors using vibration and current signals. Mechanical Systems and Signal Processing **20**(2), 403 – 420 (2006). DOI http://dx.doi.org/10.1016/j.ymssp.2004.10.010