This is a post-peer-review, pre-copyedit version of an article published in Proceedings 15th IEEE International Conference on Industrial Informatics (INDIN), Emden, Germany, 24.-26. July, 2017 The final authenticated version is available online at: http://dx.doi.org/10.1109/INDIN.2017.8104782.

Towards a Big Data Platform for Managing Machine Generated Data in the Cloud

Nicolas Ferry*, German Terrazas†, Per Kalweit‡, Arnor Solberg*, Svetan Ratchev†, Dirk Weinelt‡
*{nicolas.ferry, arnor.solberg}@sintef.no, SINTEF Digital, Norway
†{german.terrazas, svetan.ratchev}@nottingham.ac.uk, University of Nottingham, UK
‡{per.kalweit, dirk.weinelt}@tagueri.com, Tagueri AG, Germany

***Abstract*—Industry 4.0 proposes the integration of the new generation of ICT solutions for the monitoring, adaptation, simulation, and optimisation of factories. With the democratization of sensors and actuators, factories and machine tools can now be sensorized and the data generated by these devices can be exploited, for instance, to optimize the utilization of the machines as well as their operation and maintenance. However, analyzing the vast amount of data generated is resource demanding both in term of computing power and network bandwidth, thus requiring highly scalable solutions. This paper presents a novel big data platform for the management of machine generated data in the cloud. It brings together standard open source technologies which can be adapted to and deployed on different cloud infrastructures, hence reducing costs, minimising deployment difficulty and providing on-demand access to a virtually infinite set of computing, storage and network resources.**

I. Introduction

Advanced digitalisation within factories combined with information and communication technologies (ICT) results in a vision of large-scale production comprising modular, autonomous, embedded intelligence and efficient manufacturing systems. This refers to Industry 4.0 which is the integration of a wide range of concepts involving smart factory, cyber-physical systems, self-organisation, distribution, adaptation, sustainability and resource-efficiency [1][2]. The MC-SUITE project proposes a new generation of ICT enabled manufacturing process simulation and optimisation in terms of a cloud-based system that intertwines physical measurements and monitoring aimed at reducing the gap between virtual modelling and real physical processes. The interplay between the six MC-SUITE modules aims to address predictive maintenance and productivity improvement, hence transforming the manufacturing industry by reducing manufacturing process flaws, waste and variability to dramatically improve product quality and increase yield (see Fig. 1). Two of these modules are the MC-Monitor and the MC-Analytics. The first one is a retrieve-and-store platform dedicated to collect, pre-process, transmit and store continuously generated machining processes data in the cloud. The second module is a knowledge extraction platform dedicated to mine large data sets of manufacturing processes in the cloud.

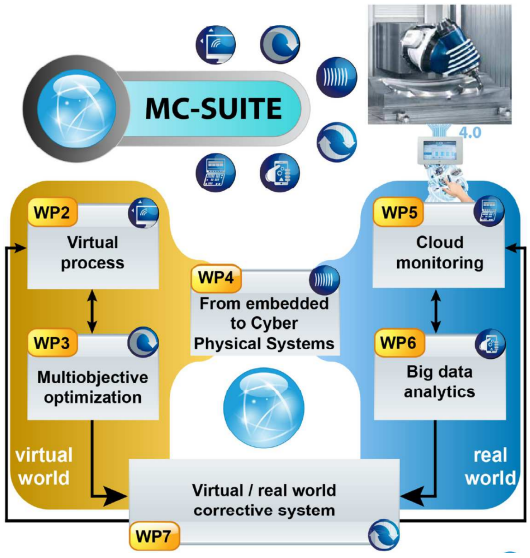

Fig. 1. Conceptual architecture of the MC-SUITE research project and its six-modules. Image by IDEKO (http://www.mc-suite.eu).

All the technological tools needed for the design and implementation of Cyber-Physical System (CPS) such as sensors, cyber-physical devices, big data and distributed infrastructures are already available. In fact, these have been applied in the context of ubiquitous manufacturing defined as the use of information technology as part of the manufacturing domain [3][4][5][6]. However, to put these technologies together it might be needed customisation for the manufacturing domain, standardisation, integration and communication architectures as well as control algorithms besides the willingness from manufacturing sites [7]. This paper focuses on data characterisation, data management challenges and technologies integration which as a whole enable a solid problem domain identification and the technology baseline needed to deliver novel manufacturing data-driven services.

Across the following sections, we showcase our work emphasising on different yet related aspects of big data management – i.e., ways to organise shop-floor data to extract information – such as data characterisation, data organisation, data reduction, restructuring and compression. In particular, Section III enlarges on the characterisation of sensory data and the challenges it brings so far. Section IV presents an architecture for shop-floor data management together with a state-of-the-art technology solution. Finally, Section V analyses some of the problems and solutions found during the modelling and data preparation stages.

II. Related Work

The activities in the manufacturing domain are driven by events usually collected through sensors and actuators. Events can be actions, activities or monitored parameter changes that influence the operational status of manufacturing operations.

From the data management perspective, the work in [8] proposes a twofold data classification: according to the occurrence period, namely low-frequency and high-frequency, and according to the source, namely men, materials, methods or machines. With such classification at hand, it is argued that machine data is actually collected in different frequencies. Thus, generating different timestamps and, henceforth, making it difficult for data linking tasks. Therefore the authors present a formalisation and bespoke algorithms to arrange and integrate data sources in terms of timestamp operations. In order to prove this, they use 14 million relational database records of data collected at a shop-floor. Although it is an elegant data management approach for one machine, nothing is said about scalability when dealing with collection and management of large data sets or data coming from the engine directly as a stream.

Five categories needed to transform current factories into intelligent sites have been discussed in [9]. The authors propose that efficient implementations should come from big data and cloud technologies, although it is argued that such technologies do not sufficiently focus on machine generated data, that they are chosen according to cost and deployment difficulty, and that in general they handle proprietary communication protocols. The work focuses on a self-awareness and self-maintenance method for industrial big data environment. In particular, showcasing adaptive clustering for machine health awareness analytics in terms of unsupervised learning algorithms. To prove this, they assess and predict a diesel engine health by collecting parameters at key operating points in order to identify patterns in data by applying a classification model.

The authors of [10] characterise a manufacturing system in terms of connection, cloud, content, community and customisation concepts. In this scope they argue that it is key to have the right information for right purposes at the right time and that having data without the right context and meaning is useless. Therefore, they idealise a framework for a predictive manufacturing system that extracts and harvests data from vibrations, pressure, etc. The conceptual architecture employs proprietary protocols, data is processed with bespoke software components including signal processing, feature extraction, health assessment, performance prediction and fault diagnosis.

A conceptual framework for the development of predictive manufacturing CPS that includes capabilities for internet of things, complex event processing and big data analytics has been described in [11]. The authors define MCPS as the marriage of manufacturing and CPS systems envisioned to handle operations in the physical world with simultaneously monitoring in the cyber world in terms of advanced data processing and simulation models for manufacturing processes. In particular, they enlarge on the different types of data generated in manufacturing domain and set up the type of technology that would enable analytics according to the seven requirements proposed for big data processing [12].

On the one hand, there is a countless number of research in the area of virtualisation and cloud-based services for manufacturing systems and the use of big data analytics. Most of them proposing either agent-based error prone architectures incapable to handle real big data scenarios or tailored to proprietary protocols, which are difficult to integrate within a scale and flexible system. In addition, there are modelling guidelines and conceptual architectures of all sorts giving solid grounds for thoughtful discussions but lacking implementations or descriptions in terms of big data management. On the other hand, manufacturing application of big data are lagging in penetration and diversity compared to other domains. This is due to the lack of adoption of standard technologies, the variety of proprietary protocols and non scalable tailored solutions. A proposal to address these follows in the next sections of our work.

III. Shop-floor Data Characterisation

One of the several sources of data of MC-SUITE are manufacturing shop-floor machines embedded with a large variety of sensors, the values of which are read, approximately, every second. These sensors capture many machining processes attributes called measurements such as spindle rates, feed rates, part

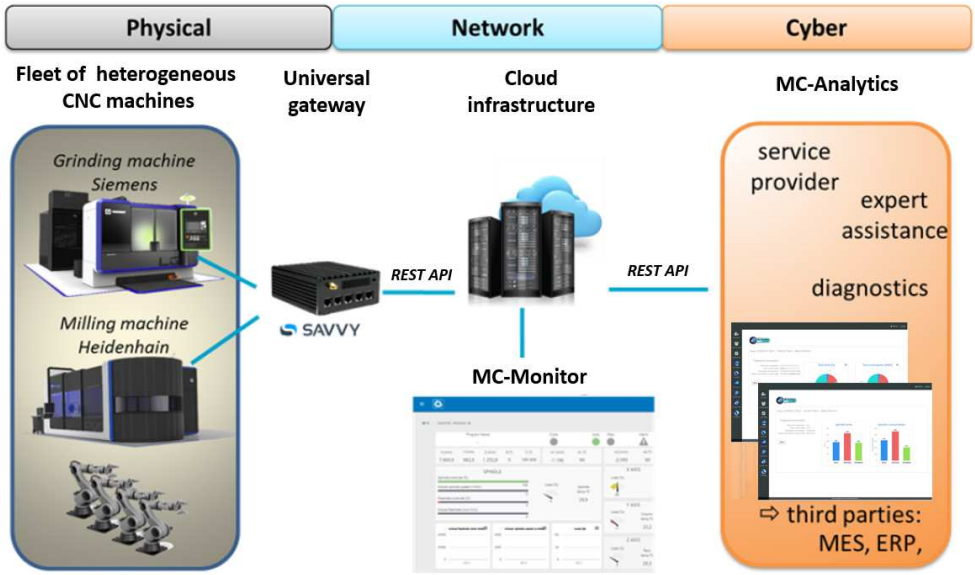

Fig. 2. The Savvy gateway serves as a sensor manager from where shop-floor data is fetch and transmitted to the MC-Monitor for processing, storing and streaming. MC-Analytics access to sensory data to provide analytics and expert assistance. Image by MC-SUITE newsletter 2.

programs, power consumption, block numbers, alarms and operators annotations to name a few. Additionally, some shop-floor machines are also equipped with video and audio devices capturing and streaming image and sound of manufacturing processes being conducted. This type of generated data is called thin data because it is a very little amount of information per device (blip of information) but potentially thousands of devices being polled on a frequent rate. Thin data goes one direction, that is from the sensors to a network, and it is in our best interest to figure out how to adequately administer it – i.e., to process, store and communicate – while it is still manageable. Manufacturing shop-floors data can be characterised in terms of:

Variety. The sensory data as well as the video and sound signals define a plethora of data types categorised into structured data, i.e. information with a high degree of organisation, semi-structured data, i.e. information that contains tags or other markers to separate semantic elements, and unstructured data, i.e. information that has neither a pre-defined data model or organisation.

Velocity. Both the sensory data as well as the video and sound signals can be generated at such a high frequency that it needs to be collected, pre-processed and stored in a framework capable of handling data in real time before they can be used in an effective way while keeping integrity, resilience, persistence and security at the required levels.

Volume. Manufacturing activities can be a complex and highly process-oriented operations. The high-frequency data generated from the sheer number and complexity of sensorized manufacturing activities result in a nonlinear vast quantity generation of information that requires a fast and efficient data management approach.

High velocity, large volume, and heterogeneous sets of data could become so large and complex that traditional data management and data processing applications result inadequate for revealing insight. Thus, MC-SUITE should be ready to deliver effective, efficient and sophisticate functionality, the challenges of which include data capture, transfer, storage, analysis, sharing, querying and updating. In order to address these, we have designed and developed a distributed, resource-efficient, highly scalable solution across shop-floor machines, MC-Monitor and MC-Analytics as depicted in Fig. 2.

In order to deliver cost-effective data analysis in MC-Analytics we should ensure an optimal ingestion and transfer of data from the shop-floor. For instance, some analytical tasks may necessitate the use of specific data sets. Our solution ensures the availability of and the access to the right shop-floor data, therefore enabling systematic data access and, consequently, efficient knowledge extraction in MC-Analytics. This has been achieved by much design and implementation of data management operations that address key pre-processing challenges such as missing data, data

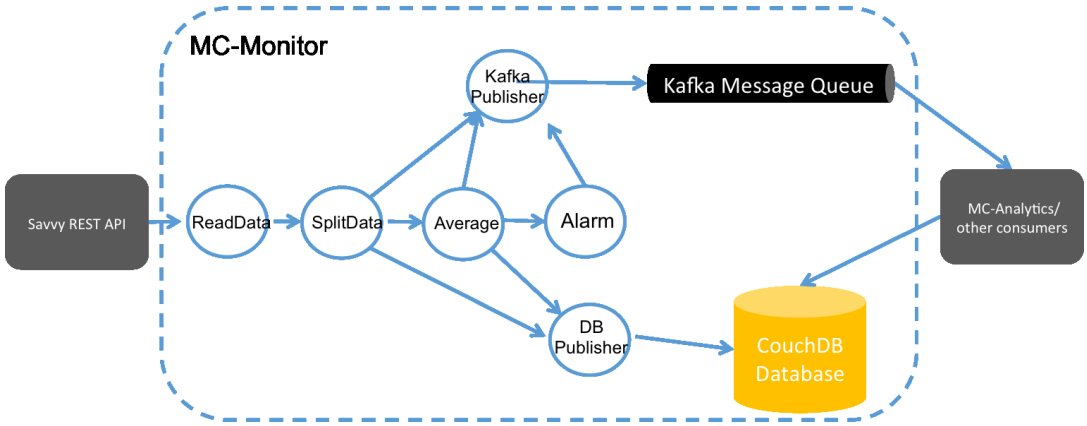

Fig. 3. Overview of the MC-Monitor Apache Storm acyclic topology comprising spout and bolts. The *ReadData* spout collects data from the Savvy REST API which is then passed to the *SplitData* bolt. This splits and transmits the data in tuples to the *KafkaPublisher* bolt, the *DBPublisher* bolt and the *Average* bolt. The *Alarm* bolt works together with the *Average* bolt to alert when a certain measurement falls outside average. The implementation is available at https://github.com/nicolasferry/vsepml.

repetition and data dimension:

Missing data. Two of the most common challenges found in shop-floor data collection relate to missing data values and file structures. Degradation or failure in the sensors might pass inaccurate and wrong readings. In addition, missing values can appear due to error in transmission of data to remote locations.

Data repetition. Highly repetitive data relates to the nature of manufacturing. That is, during a manufacturing process, the frequency at which sensors are read is likely to generate large series of repeated values. Therefore, leading to a higher than necessary volume of data being transferred.

Data size. Unstructured data such as video and sound could present challenges related to efficient transmission of data. Therefore, appealing to the use of reliable methods for signal compression and filtering techniques to only generate data when very well defined changes on the video or sound take place.

IV. Shop-floor Data Management

This section presents an architecture for shop-floor data management needed to deliver novel data-driven services in the MC-Analytics module. The first subsection describes the MC-Monitor module functionality and implementation. The second subsection depicts data formats and storage technology employed for managing both unstructured and structured data for offline analytics. The third subsection enlarges on data formats and streaming technology needed for managing structured data for online analytics.

A. The MC-Monitor

The first activity performed by big data systems consists in collecting data from the data sources. This can be divided into (a) gathering the data from the shop-floor and (b) pre-processing and providing access to the collected data. MC-SUITE data sources comprise Siemens, Heidenhain and Fidia computer numerical control (CNC) machines embedded with a large variety of machining sensors, the values of which are read by an advanced monitoring system called Savvy Smart Box[1]. This system retrieves, packs and transmits the sensory data to a cloud-based platform called Savvy Industrial Cloud via its machine-to-cloud (M2C) protocol. It is then the Savvy Industrial Cloud platform which ultimately provides MC-Monitor access to the data throughout a representational state transfer (REST) API. It is at this point where MC-Monitor pre-processes and provides access to the collected sensory data by either publishing it in a data store or keeping it in motion in the form of streams. The MC-Monitor leverages the Apache Storm[2] framework the logic of which is specified in terms of an acyclic graph topology where nodes represent sources of streams (spouts) or data processing components (bolts), and the edges represent streams of data. Fig. 3 depicts the MC-Monitor architecture designed in terms of the *ReadData* spout, the *SplitData* bolt, the *Average* bolt, the *Alarm* bolt, the *KafkaPublisher* bolt and the *DBPublisher* bolt. The *SplitData* spout reads sensory data from the Savvy REST API and transmits it to the *SplitData* bolt who splits the sensory data into streams of tuples comprising sensor name, measurement, timestamp and an

[1]http://www.savvydatasystems.com/advanced-monitoring
[2]http://storm.apache.org

unique identifier. Each of this tuples are then fed into the *KafkaPublisher*, *DBPublisher* and *Average* bolts. The first one keeps the tuples in motion by sending them to a Kafka message queue which in turn publishes the data in specified topics. The *DBPublisher* creates and publishes batches of twenty tuples into a NoSQL data base. The *Average* and *Alarm* bolts work together creating a mechanism to alert and notify when a certain measurement falls outside average. MC-Monitor spout and bolts can be instantiated several times, thus enabling parallel data processing. However, since bolts are stateless and share no memory, the routing of tuples is of particular importance, e.g. measurements from the same sensor should be transmitted to the same *Average* bolt. In order to address this, the fields grouping strategy [13] has been implemented to ensure that all tuples within a specific field are routed towards the same instance.

B. Offline Data Management

A cloud storage is used to store all relevant shop-floor data published by the *DBPublisher* bolt. We have chosen Apache CouchDB[3] since it is schema-less, supports structured as well as unstructured data, it is horizontally scalable and it exposes a simple HTTP REST interface.

The attributes of the machines in a shop-floor are stored in a CouchDB database named “MachinesList”. Thus the information related to each machine is stored as a JSON (JavaScript Object Notation) document comprising name, identification and shop-floor physical location. Given the sheer volume of data collected and in order to facilitate multiple types of analysis in MC-Analytics, different views can be associated per machine. Each of this views is nothing else than a group of sensory values captured for especific purposes in individual CouchDB databases the names of which are also listed in the machine JSON document (see Listing 1). Thus, this hierarchical structure enables the dynamic addition and deletion of machines and their views without impacting other databases.

Listing 1
Attributes of a machine in a shop-floor

```
{
  "databases":[
    {
    "name":"machine1",
    "id":"E15L17_VCVFQY_1",
    "location":"E15L17",
    "databases":[{"name":"machine1"}]
    }
  ]
}
```

[3]http://couchdb.apache.org/

The operator descriptions are stored in a CouchDB database named “Comments”. Each description is stored as a JSON document comprising the identifier of the machine operator who originates the annotation (owner), date and time when the observation has been published (timestamp in EPOCH), natural language annotation written by the operator (val), the importance level of the comment, possible values are low, moderate or high (importance), whether the observed event was planned or not (occasion), when the observed event has started (from) and when the observed event has ended (to) as shown in Listing 2.

Listing 2
An operator description

```
{
  "_id":"67ac9d997252b7c006bdcce4c000029e",
  "_rev":"1-5b5bf654f2c1e07047bb0299fb7cbffb",
  "owner":"test",
  "timestamp":"1459801408682",
  "val":"Coolant problem",
  "importance":"high",
  "occasion":"not planned",
  "from":"",
  "to":""
}
```

Sensory data is stored in a CouchDB database named after the machine it belongs to. In particular, data gathered at a given point in time is captured altogether in a single JSON document structured as a list of individual measurements. Each measurement comprises a sensor identifier, value of the sensor reading, type of information, unit of the value and a coefficient (see Listing 3). This JSON document may also contain an extra field generated by the sparsity data mechanism explained in Section V. This field is called DocumentSkipped and its value indicates the amount of previously skipped identical measurements.

Listing 3
A machine sensory data

```
{
  "_id":"1451692802000",
  "_rev":"1-ca46f3b012d07e31ed777bc97fa95863",
  "DocumentSkipped":891,
  "Measurements": [
    {
    "Measurement":"1547410",
    "SensorID":"Axis_X",
    "Type":"positionActualWCS",
    "Unit":"mm",
    "Coeff":"d1000"
    },
    {
    "Measurement":"46",
    "SensorID":"Axis_X1",
    "Type":"power",
    "Unit":"percent",
    "Coeff":""
    }
  ]
```

```
}
```

C. Online Data Management

A message queue is used to stream all relevant shop-floor data published by the *KafkaPublisher* bolt. We have chosen Apache Kafka[4] since it offers the most promising performance in terms of message publication and consumption, it is horizontally scalable and fault tolerant, it provides space and time decoupling, and it exposes a simple HTTP REST interface with the capability to navigate and re-read messages. Thus, the *KafkaPublisher* bolt publishes sensory data to one or more topics. A topic is a queue where one or multiple subscribers can register to listen and read data of interest. It is worth noting that topics can be created dynamically facilitating scalability and flexibility of the MC-Monitor and MC-Analytics. Currently, the following four topics have been created:

SensorsChunk. Is a topic defined for messages grouping data generated by all sensors of a given machine at a specific time.

Sensor. Is a topic created for messages containing data generated by individual sensors.

Alarms. Is a topic created for messages containing data generated by an alarm. This refers to a mechanism that compares the average value of a sensor to the current one and raises an alarm if the discrepancy between these two is too high.

Average. Is a topic defined for messages containing the average value generated from the last n measurements of a given sensor.

Listing 4
Message published in the Sensor topic

```
{
  "_id":"1451957699000",
  "MachineID":"E15L17_VCVFQY_1",
  "SensorId":"Axis_Y",
  "Measurements":"30",
  "Type":"motor",
  "Coeff":"degreeCelsius",
  "Unit":"temperature"
}
```

The structure of the messages published in the SensorsChunk topic is depicted in Listing 3, i.e. an array of measurements read from individual sensors at a given point in time comprising sensor identifier, value, type, coefficient and unit. The structure of a message published in the Sensors topic comprises the machine identification, sensor identifier, the value of the sensor reading, type of information, unit of the value and a coefficient (see Listing 4). The content of the messages published in the Averages and Alarms topics is similar to the one above but with an additional field for the average.

[4]https://kafka.apache.org/

V. Data Enhancements

The success of a data analytic service involves far more than choosing or developing an algorithm and running it over data. In most cases, results can be improved by a suitable choice of input parameter values and the appropriate choice of the data at hand. The last one constitute a kind of data engineering, that is engineering the input data into a form suitable for analytics and to make it more effective. Thus, the restructuring of the sensory data is performed by the components described in Section IV. In order to best fit the data management requirements for MC-Analytics, this components were enhanced with a context-aware data collection mechanism that dynamically adapts to factory changes, a feature to retrieve-and-resource relevant sensory data attributes and a size reduction feature to optimise sensory data management. The last two aiming at improving the performance and scalability of the MC-Analytics by reducing the size and number of messages exchanged which, as a result, optimise the volume of data in CouchDB databases.

A. Context-Aware Data Retrieval

As explained in Section IV-A the *ReadData* spout has access to shop-floor data throughout the Savvy Industrial Cloud REST API. However, the format and content of this data is insufficient for knowledge extraction tasks in MC-Analytics. Indeed, as depicted in Listing 5, the JSON document provided by the Savvy REST API, key semantic information such as the sensors name and type as well as the unit of the measurements are missing.

Listing 5
Savvy REST API output

```
{
  "machine":"E15L17_VCVFQY_1",
  "group":"QE1KWH",
  "data":
    {
    "timestamp":
      {
      "date":"2017-01-20T10:22:14.286Z"
      },
      "B3JCPQ":"10",
      "A5TBSV":"0"
    }
}
```

Therefore, we have enriched the *ReadData* spout with the capability to perform HTTP requests to collect not only information about shop-floor machines but also information about data and metadata of the machine sensors. Once the details about machines and sensors are retrieved, the *ReadData* spout uses this information to register with the Savvy REST API and initiate the continuous transfer of data all of which is subsequently re-formated by the *SplitData* bolt. For example, Listing 6 shows HTTP requests to collect data and metadata such as list of machines available for monitoring in location E1L1, a list acquisition groups associated to machine E15L17_VCVFQY_1 and a list of sensors of the CZDY1B acquisition group. This information is then combined to perform a final HTTP request that streams sensory data from machine E15L17_VCVFQY_1. As a result, the elements of this stream are JSON objects composed by measurements collected from all sensors at a given point in time (see Listing 3).

Listing 6
HTTP requests to the Savvy REST API

```
GET /locations/E1L1/machines
GET /v1/locations/E15L17/machines/E15L17_VCVFQY_1/groups
GET /v1/locations/E15L17/machines/E15L17_VCVFQY_1/groups/
    CZDY1B/sensors
GET /v1/stream?track=E15L17_VCVFQY_1
```

Enriching the *ReadData* spout for leveraging the Savvy REST API improves the flexibility of the entire topology since the machines and sensors information can now be obtained dynamically. That is, the shop-floor equipment can change (e.g. new sensors can be added to or removed from the machines) and the whole mechanism for retrieving data will reconfigure itself, hence adapting to factory level modifications without substantial modifications.

B. Sensory Data Attribute Selection

In data mining, there is often far too many data attributes to handle and some of them could become clearly irrelevant or redundant. Hence, applying dimensionality reduction yields a more compact, more easily interpretable representation for the objectives, thus helping focus the attention on the most relevant pieces of information. Although there exists automatic methods, the best way to proceed with attribute selection in our case is manually since we have a deep understanding of what the attributes actually mean. Therefore, in order to provide the MC-Analytics with the most relevant data, the *SplitData* bolt transforms the data supplied by the Savvy system (via the *ReadData* spout) into another JSON object. This is done by setting a descriptor that specifies the names of the sensors of interest (see Listing 7 for an example). The transformation and restructuring is then carried out by the *SplitData* bolt as defined in Algorithm 1.

input : data and metadata JSON objs from ReadData spout ($objD$ and $objMD$)
output: new JSON obj with relevant measurements ($newObj$)

$newObj$ = new JSON object;
$newObj.measurements$ = new JSON list;
foreach $objD.measurements[i]$ **do**
  $name$ = $objMD.measurements[i].name$;
  if *name is in descriptor* **then**
    $fields$ = $objMD.measurements[i]$;
    $newMsrnt$ = new measurement;
    $newMsrnt.name$ = $fields.name$;
    $newMsrnt.val$ = $objD.measurements[i].val$;
    $newMsrnt.type$ = $fields.type$;
    $newMsrnt.unit$ = $fields.unit$;
    $newMsrnt.coeff$ = $fields.coefficient$;
    add $newMsrnt$ to $newObj.measurements$;
  else
    $objD.measurements[i]$ is ignored
  end
end
return $newObj$;

Algorithm 1: Restructuring sensory data task within *SplitData* bolt

Listing 7
Descriptor for power consumption analisys in MC-Analytics

```
Cnc_Program_Name_RT
Cnc_Program_BlockNumber_RT
Axis_X1_power_percent
Axis_X2_power_percent
Axis_Y_power_percent
Axis_Z_power_percent
Cnc_Tool_Number_RT
Cnc_IsAutomaticModeActive
Cnc_IsCycleOn_RT
Spindle_IsAutomatic
Spindle_IsDirect
Spindle_IsOrtogonal
Spindle_positioning
Spindle_Power_percent
Axis_X_positionActualMCS_mm_d1000
Axis_Y_positionActualMCS_mm_d1000
Axis_Z_positionActualMCS_mm_d1000
Axis_FeedRate_actual
Cnc_IsManualModeActive
```

C. Sparsity Induced Sensory Data

The sensory data comprises very little amount of information per sensor (potentially thousands) being polled on a frequent rate (approximately every second). One of the management issues this presents is dealing with repetitive information, i.e. repeated data values coming from a given sensor during a certain period

of time (perhaps tens of minutes). Ignoring repetitive data values could not only implicate less traffic of unnecessary data but also dramatically reduce the space capacity within the cloud storage. Therefore, one of the ways to deal with this is to convert sensory data into sparse data, i.e. capturing new values only when they change. Here we outline a simple method to do this.
In addition to the sensory data attribute selection described earlier, and before adding the list of measurements into a JSON object, the *SplitData* bolt checks whether the provided set of measurements is similar to the one processed right before. This is simply done by comparing all sensor values collected at time t to the values collected at time $t-1$. If both sets are equal, this measurement is not added to the JSON object generated by the bolt and an associated counter is incremented. As soon as the measurement starts to be different, the value of the counter is set to the DocumentsSkipped field of the JSON object to indicate how many measurements were skipped. Therefore, giving the origin to sparsity-inducing dictionaries associated to the data. The features obtained from sparsity based representation provide discriminative information useful for analytics based on classification. In addition, the sparsity of a signal implemented in terms of compressed sensing [14] can be exploited by data analytics that need the entire signal since this can efficiently be recovered from far fewer samples than required [15].

VI. Conclusion and Further Work

This work focused on data characterisation, data management challenges and the implementation of a shop-floor data management platform that integrates the standard open source big data technologies Apache Storm, Kafka, CouchDB and JSON. This infrastructure is specified using Cloud Modelling Language (CloudML[5]) which, in turn, can be automatically deployed on and adapted to different cloud solutions using CloudMF [16], hence reducing cost and deployment difficulty. In addition, it works at the infrastructure-as-a-service level ensuring that we have control over the protocols. We have showcased our work emphasising on different yet related aspects of big data management such as data characterisation, data organisation, data reduction, restructuring and compression. In terms of shop-floor data, we are currently looking at the identification and collection of machine alarms and exploring the management of video and audio data which is stored in an open-source Dropbox-like service called Owncloud. Since cyber security penetration in manufacturing domain is an unaddressed need across researchers and practitioners we will be exploring the best standard approaches to address vulnerability as well as to add resilience test to cope with complex events and respond in acceptable time as well as sustain operations in face of disturbances.

[5]http://cloudml.org

Acknowledgment

The authors would like to thank the support of the Horizon 2020 MC-SUITE (ICT Powered MaChining Suite) project funded by the European Commission under grant agreement Nº 680478.

References

[1] H. Lasi, P. Fettke, H.-G. Kemper, T. Feld and M. Hoffman, *Industry 4.0*. Business & Information Systems Engineering, 6(4), pp. 239–242, 2014.

[2] Federal Ministry of Education and Research, Project of the future: Industry 4.0, http://www.bmbf.de/en/19955.php, 2013.

[3] L. Ferreira, G. Putnik, M. Cunha, Z. Putnik, H. Castro, C. Alves, V. Shah and M.L.R. Varela, Cloudlet architecture for dashboard in cloud and ubiquitous manufacturing, CIRP 12, pp. 366–371, 2013.

[4] J. Kiirikki, M. Haag, Ubiquitous assembly cell concept and requirements, CIRP 12, pp. 157–162, 2013.

[5] K.C. Lee, N. Chung, J. Byun, Understanding continued ubiquitous decision support system usage behavior, Telematics and Informatics, 32(4), pp. 921–929, 2015.

[6] I. Horvath, R.W. Vroom, Ubiquitous computer aided design: A broken promise or a Sleeping Beauty?, Computer-Aided Design, vol. 59, pp. 161–175, 2015.

[7] A. Botta, W. de Donato, V. Persico, A. Pescapé, Integration of Cloud computing and Internet of Things, Future Generation Computer Systems, 56(C), pp. 684–700, 2016.

[8] R. Kim, S. Chi and W.C. Yoon, Data integration and arrangement in the shop floor based-on time stamp: An illustrative study of CNC machining, 8th International Conference on Ubiquitous and Future Networks, pp. 121–124, 2016.

[9] J. Lee, H.-A. Kao, S. Yang, Service Innovation and Smart Analytics for Industry 4.0 and Big Data Environment, 6th CIRP Conference on Industrial Product Service Systems, CIRP vol. 16, pp. 3–8, 2014.

[10] J. Lee, E. Lapira, B. Bagheri, H. Kao, Recent advances and trends in predictive manufacturing systems in big data environment, Manufacturing Letters, 1(1), pp. 38–41, 2013.

[11] R. F. Babiceanu, R. Seker, Big Data and virtualization for manufacturing cyber-physical systems: A survey of the current status and future outlook, Computers in Industry, 81(C), pp. 128–137, 2016.

[12] K. Krumeich, D. Werth, P. Loos, J. Schimmelpfennig, S. Jacobi, Advanced planning and control of manufacturing processes in steel industry through Big Data analytics: Case study and architecture proposal, IEEE International Conference on Big Data, pp. 16–24, 2014.

[13] A. Toshniwal, S. Taneja, A. Shukla, K. Ramasamy, J.M. Patel, S. Kulkarni, J. Jackson, K. Gade, M. Fu, J. Donham, N. Bhagat, S. Mittal, D. Ryaboy, Storm@twitter, ACM SIGMOD International Conference on Management of Data, pp. 147–156, 2014.

[14] R. Rubinstein, A.M. Bruckstein and M. Elad, Dictionaries for Sparse Representation Modeling, Proceedings of the IEEE, 98(6), pp. 1045–1057, 2010.

[15] D.L. Donoho, Compressed sensing, IEEE Transactions on Information Theory, 52(4), pp. 1289–1306, 2006.

[16] N. Ferry, H. Song, A. Rossini, F. Chauvel, A. Solberg, CloudMF: Applying MDE to Tame the Complexity of Managing Multi-cloud Applications, IEEE/ACM International Conference on Utility and Cloud Computing, 269–277, 2014.