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, analysing 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 and driven by elicited use case requirements related to energy consumption, production evaluation and machine vibrations. The first use case is a descriptive analysis that reports on the kilowatts per hour a computer numerical controlled (CNC) machine consumed during a given period of time at different levels such as part program, tools, spindles and machine axes. The second is a classification analysis that reveals the percentage of time a CNC machine has spent per day either machining a part piece, being setup by operators, stopped or doing other operations. The last use case is a regression analysis that sheds light on whether vibrations captured along a CNC machine axes during machining could be considered as chatter since this can potentially impact tool wear and part finishing.

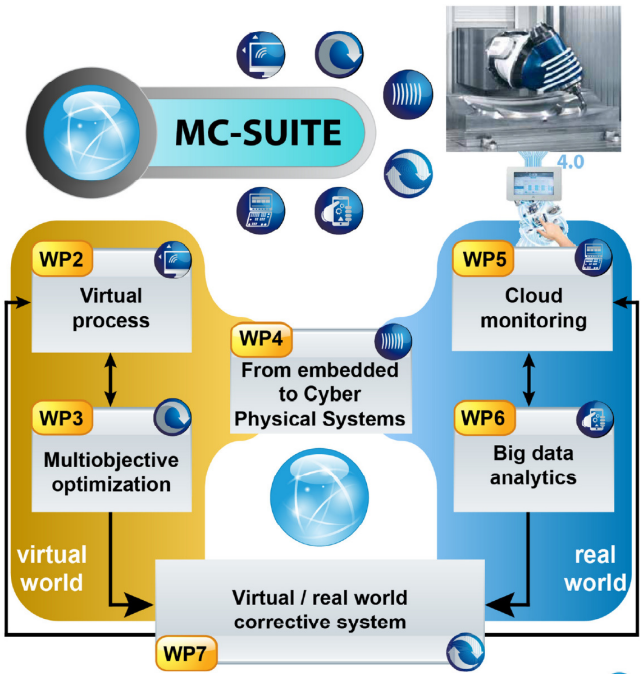

Fig. 1. Conceptual architecture of the MC-SUITE research project and its six-modules. Image by IDEKO (http://www.mc-suite.eu).

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, bringing these technologies together entails customisation for the manufacturing domain, standardisation, integration, communication architectures as well as control algorithms besides the willingness from manufacturing

sites [7]. Hence, 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 controlled through sensors and actuators. These events can be actions, activities or monitored parameter changes that influence the 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 humans, materials, methods or machines. the work argues 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 and in order to prove this approach 14 million records of data collected at a shop-floor are used. Although it is an elegant data management solution for one machine, nothing is said about scalability when dealing with collection and management of large data sets or data as streams.

Manager and operator interactions, machine fleet, product and process quality, big data technologies, and sensor networks have been defined as key categories where intelligent data exploitation is needed to transform current factories into smart sites [9]. 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 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 they generally handle proprietary communication protocols.

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 the 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 and 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 propose to classify manufacturing data in terms of operational, time-based, supply chain, physical property, tracking, social networking, customer service and marketing types of data. In addition, they set up modelling guidelines to enable planning and control of manufacturing operations aligned to the seven architectural requirements proposed for big data processing [12]. These requirements refer to online and offline processing, detection and filtering of complex events, real-time big data storage capabilities, adaptive prediction models, proactive incident handling, intelligent process planning and control, and proactive process decision making. As a whole this combination results in a solid and sound approach, however the work is only limited to deliver well structured modelling guidelines for development without an actual demonstration with technologies.

Three different data acquisition and management architectures have been defined in [13] to address process monitoring, data analysis and fault detection in industrial and agricultural processes. In particular, the authors have equipped agricultural harvesters with data acquisition units capable of transmitting nearly 3000 data attributes to a back-end platform via Apache Kafka. The back-end platform comprises Apache Storm server for fast data processing and deposition into distributed batch processing technologies that, ultimately, serve data to a variety of analytics modules. Similar arrangements of these technologies have been proposed in [14] and [15]. The first one focuses on ingestion, aggregation, buffering and lightweight stream analysis of milling machines logs. The last one focuses on milling machines anomaly detection in terms of energy consumption data sets analysis. Although these works have successfully addressed data collection and aggregation across production environments, they still lack generalisation, do not facilitate cross vendor deployment solutions, miss data characterisation and pre-processing stages to fully understand the domain. In addition, they focus on the creation of bespoke solutions rather than delivering a general, reusable, open source approach. On the one hand, there is a large amount of research in the area of virtualisation and cloud-based services

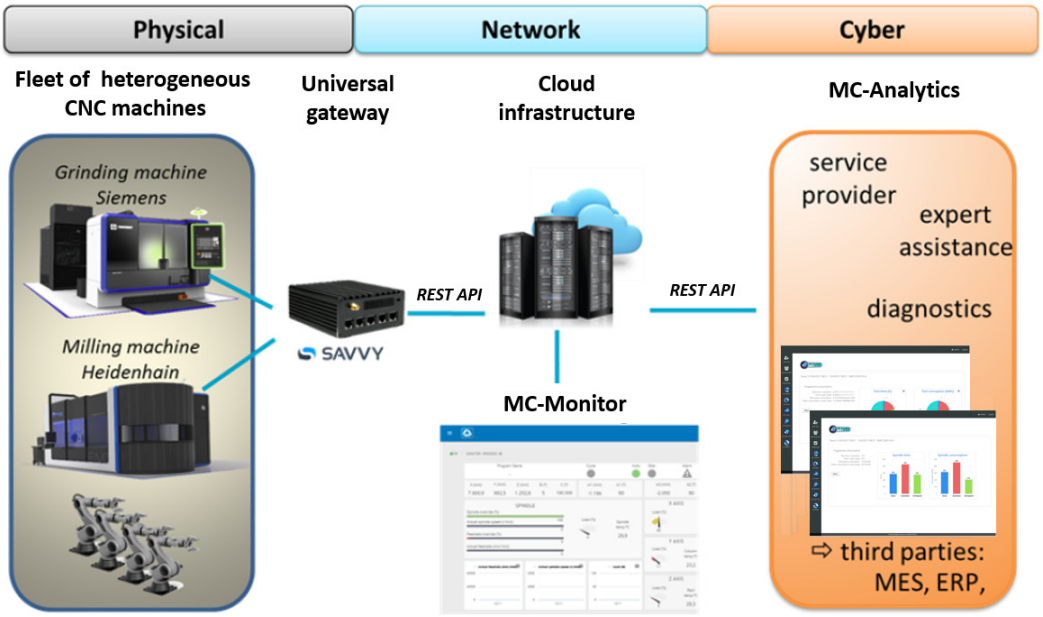

Fig. 2. The Savvy gateway serves as a sensor manager from where shop-floor data is fetch and transmitted to the MC-Monitor for processing, storing and streaming. MC-Analytics access to sensory data to provide analytics and expert assistance. Image by MC-SUITE newsletter 2.

for manufacturing systems and big data analytics. Most of these works propose frameworks that work well in specific scenarios or are tailored to proprietary protocols which become difficult to integrate within a scalable and flexible solution. 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 [16]. 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. Hence, 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 in MC-SUITE are manufacturing shop-floor CNC 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 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 for capturing and streaming image and sound of the manufacturing processes being conducted. The 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 complex and highly process-oriented operations. The high-frequency data generated from the sheer number of sensorized manufacturing activities could result in a nonlinear vast quantity generation of information that demands 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, reliable, highly scalable solution across shop-floor machines, the MC-Monitor and the MC-Analytics as shown in Fig. 2.

In order to deliver cost-effective data analysis in the MC-Analytics we should ensure an optimal ingestion and transfer of data. For instance, energy consumption analysis necessitates specific sensory data compared to machine vibrations. Thus, 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. This has been achieved by design and implementation of data management operations that address key pre-processing challenges such as missing data, data repetition and data size:

Missing data. Two of the most common challenges found in shop-floor data collection relate to missing data values and file structures. This could be due to degradation or failure in the sensors which 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 signal compression methods and filtering techniques to generate data only when very well defined changes on image or sound take place.

IV. Shop-floor Data Management

This section presents an architecture for shop-floor data management needed to deliver 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 for managing structured data for online analytics.

A. The MC-Monitor

The first activity performed by big data systems consists in collecting data from 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 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 [17]. 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 and makes data available through a representational state transfer (REST) API. It is at this point where the MC-Monitor fetches, 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 [18] which has been chosen due to its reliability on processing streaming real-time data. The logic of this framework 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. In this way, the *ReadData* 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 unique identifier. Each of these tuples are then fed into the *KafkaPublisher*, *DBPublisher* and *Average* bolts. The first one keeps the tuples in motion by sending them to a Apache Kafka [19] 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. The 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 [20] has been implemented to ensure that all tuples within a specific field are routed towards the same instance.

B. Offline Data Management

A cloud-based storage is used to store use case specific shop-floor data published by the *DBPublisher* bolt. We have chosen Apache CouchDB [21] since it is schema-less, supports structured as well as unstructured data, it is horizontally scalable and it exposes a native 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 the MC-Analytics, different views can be associated per CNC machine. Each of these 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 views without affecting other databases.

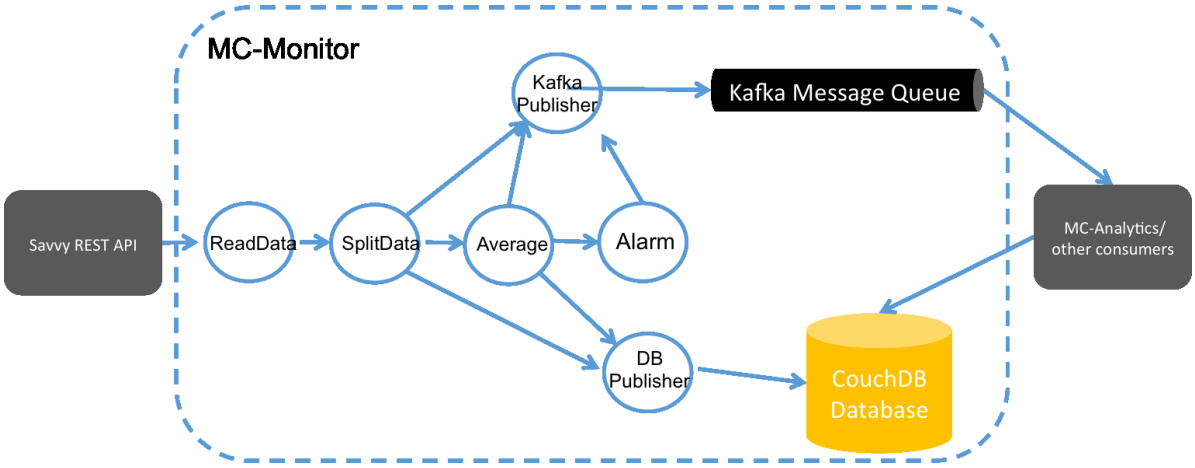

Fig. 3. Overview of the MC-Monitor Apache Storm acyclic topology comprising spout and bolts. The *ReadData* spout collects data from the Savvy REST API which is then passed to the *SplitData* bolt. This splits and transmits the data in tuples to the *KafkaPublisher* bolt, the *DBPublisher* bolt and the *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.

Listing 1
ATTRIBUTES OF A CNC MACHINE IN A SHOP-FLOOR


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


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 format), 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": "335",
    "SensorID": "Axis_FeedRate",
    "Type": "actual",
    "Unit": "",
    "Coeff": ""
  },
  {
    "Measurement": "TAKOAK_FRESATZEKO.H",
    "SensorID": "Cnc_Program_Name",
    "Type": "RT",
    "Unit": "",
    "Coeff": ""
  },
  {
    "Measurement": "167",
    "SensorID": "Cnc_Program_BlockNumber",
    "Type": "RT",
    "Unit": "",
    "Coeff": ""
  },
  {
    "Measurement": "50",
    "SensorID": "Spindle_Power",
    "Type": "percent",
    "Unit": "",
    "Coeff": ""
  },
```

```
  {
    "Measurement": "300",
    "SensorID": "Cnc_Tool_Number",
    "Type": "RT",
    "Unit": "",
    "Coeff": ""
  },
  {
    "Measurement": "-411784",
    "SensorID": "Axis_X",
    "Type": "positionActualWCS",
    "Unit": "mm",
    "Coeff": "d1000"
   }
  ]
}
```

C. Online Data Management

A message queue is used to stream use case specific shop-floor data published by the *KafkaPublisher* bolt. We have chosen Apache Kafka 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 the MC-Analytics. Currently, the following four topics have been created:

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

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

Alarms. It 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. It 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 value, identifier, type of information, unit and coefficient. The structure of a message published in the Sensors topic comprises the machine identifier, sensor identifier, value, type of information, unit and 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.

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 to conduct more efficient analytics. 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 the MC-Analytics, these 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, minimise 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. In other words, 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, the data retrieval mechanism adapts to factory level modifications, i.e. the context, without the need of substantial modifications.

Listing 7
DESCRIPTOR FOR POWER CONSUMPTION ANALISYS IN MC-ANALYTICS

```
Fecha
Cnc_Program_Name_RT
Cnc_Program_BlockNumber_RT
Axis_X1_power_percent
Axis_X2_power_percent
Axis_Y_power_percent
Axis_Z_power_percent
Systems_severity_ram_X
Systems_severity_ram_Y
Systems_severity_ram_Z
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_X_positionActualWCS_mm_d1000
Axis_Y_positionActualMCS_mm_d1000
Axis_Y_positionActualWCS_mm_d1000
Axis_Z_positionActualMCS_mm_d1000
Axis_Z_positionActualWCS_mm_d1000
Axis_FeedRate_actual
Cnc_IsManualModeActive
```

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 to a more compact and easier representation for the objectives, thus helping focus on the most relevant pieces of information. Therefore, in order to provide the MC-Analytics with data relevant to specific use cases, the *SplitData* bolt transforms the data supplied by the Savvy REST API (via the *ReadData* spout) into another JSON object. This is done by setting a use case-specific descriptor that lists the names of the sensors of interest (see Listing 7 for an example). In this way, the transformation and restructuring of data is automatically 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

C. Sparsity Induced Sensory Data

The sensory data comprises very little amount of information per sensor (potentially thousands) being polled at a frequent rate (approximately every second). One of the challenges this presents is dealing with repetitive information, i.e. repeated data values coming from a given sensor during a fixed period of time. 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. Once 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 [22] can be exploited by other data analytic services that need the entire signal since this can efficiently be recovered from far fewer samples than required [23].

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 [24]) which, in turn, can be automatically deployed on and adapted to different cloud solutions using CloudMF, hence reducing cost and deployment difficulty as discussed in [25]. 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 currently collected in an open-source Dropbox-like service called Owncloud for storage purposes only. 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.

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, CIRP 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] S. Windmann, A. Maier, O. Niggemann, C. Frey, A. Bernardi, Y. Gu, H. Pfrommer, T. Steckel, M. Krger, R. Kraus, Big Data Analysis of Manufacturing Processes, 12th European Workshop on Advanced Control and Diagnosis, pp. 136–142, 2015.

[14] J. Park, S. Chi, An implementation of a high throughput data ingestion system for machine logs in manufacturing industry, 8th International Conference on Ubiquitous and Future Networks, pp. 117–120, 2016.

[15] H. Chen, X. Fei, S. Wang, X. Lu, G. Jin, W. Li, X. Wu, Energy Consumption Data Based Machine Anomaly Detection, 2nd International Conference on Advanced Cloud and Big Data, 2014.

[16] R. S. Peres, A.D. Rocha, A. Coelho, J. Barata Oliveira, A Highly Flexible, Distributed Data Analysis Framework for Industry 4.0 Manufacturing Systems, Service Orientation in Holonic and Multi-Agent Manufacturing, pp. 373–381, 2017.

[17] Savvy Advanced Monitoring, http://www.savvydatasystems.com/advanced-monitoring. Accessed June 2017.

[18] Apache Storm TM, http://storm.apache.org/index.html. Accessed June 2017.

[19] Apache Kafka. A distributed streaming platform, https://kafka.apache.org/. Accessed June 2017.

[20] 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.

[21] Apache CouchDB relax, http://couchdb.apache.org/. Accessed June 2017.

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

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

[24] CloudML. Model-based provisioning and deployment of cloud-based systems, http://cloudml.org/. Accessed June 2017.

[25] 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.