International Conference on Industry 4.0 and Smart Manufacturing (ISM 2019)

XRepo - Towards an information system for prognostics and health management analysis

Alfonso Ardila[a], Felipe Martinez[a], Kelly Garces[a], Giacomo Barbieri[b], David Sanchez-Londono[b,*], Andrea Caielli[c], Laura Cattaneo[c], Luca Fumagalli[c]

[a]*Department of Systems and Computing Engineering, Universidad de los Andes, Carrera 1 # 18A - 12, Bogota 111711, Colombia*
[b]*Department of Mechanical Engineering, Universidad de los Andes, Carrera 1 # 18A - 12, Bogota 111711, Colombia*
[c]*Department of Management, Economics and Industrial Engineering, Politecnico di Milano, P.za Leonardo da Vinci 32, Milano 20133, Italy*

* Corresponding author. *E-mail address:* d.sanchezl@uniandes.edu.co

Abstract

In the Industry 4.0 vision, Prognostics and Health Management (PHM) is expected to assist domain experts in the generation of maintenance decisions. PHM relies on the processing of data sensed from the manufacturing plant for inferring the future performance of production systems. The acquisition and management of data brings different challenges such as data integration, heterogeneity, search usability and volume. To be best of authors' knowledge, an information system for sharing maintenance data able to fulfill the aforementioned challenges is not available yet. XRepo is proposed within this paper and faces three of the identified challenges through selected functionality: i) Heterogeneity: stored data is complaint with a standard format that includes the information necessary for performing PHM analysis; ii) Integration: data is uploaded to the repository through files or web services; iii) Search usability: stored data can be filtered by criteria and downloaded. This work is meant to be an initial effort towards the generation of a common information system for PHM analysis.

Keywords: Industry 4.0; Prognostics and Health Management; Maintenance Datasets; Information System; Maintenance Repository

1. Introduction

Current maintenance strategies have progressed from breakdown maintenance, to preventive maintenance and condition-based maintenance, and eventually towards prognostics and health management [1]. *Breakdown maintenance* is the earliest form of maintenance, where no actions are taken to maintain the equipment until it breaks and consequently needs a repair or replacement. In the 1950s, *preventive maintenance* strategies are introduced and require maintenance on a time (or usage) interval regardless of the health condition of the asset. *Condition based maintenance* (CBM) utilizes sensor readings to detect the occurrence of a fault or anomaly, isolate the location of the fault and identify the type of failure. The potential of CBM can be extended to *Prognostics and Health Management* (PHM) in which predictive tools and algorithms are adopted to infer the future performance of the system [2]. This prediction capability has many benefits. On one hand, companies can dynamically schedule maintenance in order to minimize system downtime. On the other hand, it may also enable the estimation of the location of faults guiding operators on what components to check during the maintenance activity [3].

In the last years, industry has assisted to developments in the fields of: i) *Technology*: higher availability and affordability of sensors and data acquisition systems; ii) *Internet of Things*: tether-free and connected data management platform with real-time streaming and processing capabilities; iii) *Data Analytics*: advances in predictive analytics and processing power. Industry 4.0 is defined as the integration of these developments within current industrial practices [4].

PHM research field has grown significantly due to the Industry 4.0 movement.

PHM heavily relies on data acquisition and management, and brings the following challenges:

1. Data Integration [5]: data from different sensors need to be collected and integrated for the health assessment and prediction of the state of manufacturing systems;
2. Data Heterogeneity [6]: two types of heterogeneity can occur when data are integrated: i) Syntax: data can be stored in different file formats and structures; ii) Semantics: data can present different units, contexts (normal/faulty conditions), etc. Heterogeneity makes difficult the processes of storing, analyzing, and exploiting data;
3. Data Search Usability [7]: storing simple raw data would result in unusable data since their context and their position within the repository would rely on the users' knowledge of the datasets. Another aspect of this challenge is that once the data is stored, users need to navigate this deluge of datasets that lack an ordered structure. This also results in poor usability;
4. Data Volume [8]: monitoring a manufacturing plant implies the gathering of large amounts of data since different sensors can scan the production systems at high frequency rates. This results in an unmanageable volume of data which is complex to store, process and use.

Sharing of maintenance data is a need in both academia and industry. Researchers and practitioners usually perform comparative analyses between different maintenance datasets. In fact, the comparison of data from different plants and experiments is fundamental for identifying the failure behavior of the investigated system, and for validating PHM algorithms. Nowadays, there is not a standard management of the information since in the current practice each institution creates its own repository with disparate formats; e.g. [9]–[11]. The heterogeneity of data makes the process of analyzing information from different sources non-trivial. A case study involving 20 experts from 9 different Italian companies that work on maintenance and digitalization processes confirms that the lack of collaboration mechanisms to share data between technology suppliers, service providers, and end users is one of the main barriers in the process of digital transformation of maintenance [12]. Therefore, we believe that a common platform for storing and searching maintenance datasets would enhance research in PHM.

Towards a solution that allows data exchange among research groups and companies, we propose an information system that deals with three of the aforementioned challenges, i.e., data integration, data heterogeneity and data search usability. The data volume challenge is out of the scope of this paper and are under investigation. The remaining of the paper is organized as follows: related works are described in Section 2. Section 3 shows the need of a common information system for maintenance datasets through two application examples from the academic and industrial domain. Section 5 illustrates the proposed information system, while Section 5 validates it. Eventually, conclusions and future work are reported in Section 6.

2. State of the art

In 2001, the Machinery Information Management Open Systems Alliance (MIMOSA) published the *Open System Architecture for Condition-Based Maintenance* (OSA-CBM) [13]. The OSA-CBM specifications define a standard architecture and framework for implementing CBM systems. Whereas this standard characterizes the acquisition of data, it does not represent information necessary for a PHM analysis as the operational conditions of the production plant and the labeling of the data; i.e. marking certain data values as potential indicators of faulty/normal conditions.

In [14], one the authors proposes a metamodel to allow the development of an integrated system that combines the features of Computerized Maintenance Management Systems (CMMS) - Reliability Centered Maintenance (RCM) - CBM systems. However, this metamodel does not focus on the development of an information system for the storage of data acquired for PHM analysis.

MTConnect^{TM} is a protocol designed for the exchange of data between shop floor equipment and software applications used for monitoring and data analysis [15]. Data from shop floor devices is presented in XML format and is retrieved from information providers using HTTP as the underlying transport protocol. However, MTConnect^{TM}'s purpose is to allow the sharing of data and not the generation of a structured information system for PHM-related analysis.

An information system with a purpose similar to the one proposed in this paper is the *AMBER data repository* [16]. AMBER is an online infrastructure that focuses on collecting raw maintenance data and on sharing it among researchers. However, the AMBER repository does not implement functionality for filtering the data by criteria, limiting the usability of the datasets.

Eventually, datasets available online were studied in order to identify the information that may help in solving the challenges faced within this work. Five datasets were analyzed: three consisting in failure data of rolling bearings [9]–[11], one concerning failures in the air pressure system of trucks [17], and the last about cutting blade degradation in an industrial setting [18]. A comparison concerning the information provided in each is reported in Table 1. The following notation is adopted for the comparison:

- ✓: information included in the dataset file;
- o: information not included in the dataset file, but available elsewhere (e.g. in the paper);
- x: information not available.

Table 1. Data fields used in various PHM datasets.

Data field	Dataset				
	[9]	[10]	[11]	[17]	[18]
Operative ranges	x	o	o	x	x
Experiment condition	o	o	o	o	o
Operative condition	o	o	o	o	x
Sample Start/End Time	✓	x	x	x	✓
DAQ specification	o	x	x	x	x
Sensor specification	o	o	o	x	o
Sampling frequency	o	o	o	x	o
Sensor ID	x	✓	x	x	x

The information identified in the analyzed datasets that helps in facing the mentioned challenges is:

- *Operative ranges*: the information of the operative ranges of the equipment used within the experiment can be adopted for data searching by criteria. This searching functionality helps on facing the search usability challenge;
- *Experiment description and operative conditions*: different experiments can be performed on a certain physical asset. The description of the context of the experiment is fundamental for proper PHM analysis in which different scenarios can be evaluated;
- *Per-sample date and time*: an experiment may contain samples from different sensors. Each sensor value must be presented together with its date/time for proper data integration and data fusion;
- *Data acquisition device, sensor specifications and sampling frequency*: the specification of the devices used for sensing and acquiring the data is fundamental for its correct interpretation. For example, the analog voltage/current signal of the sensor must be converted into the digital value of the corresponding physical variable. Here, the number of digits of the A/D converter is necessary for calculating the conversion factor;
- *Sensor ID*: when different sensors are used in a single experiment, the identification of which measurements were taken from each sensor is fundamental for a correct interpretation and integration of the data;
- *Measured variable and unit*: a sensor may measure more than one physical variable; e.g. some accelerometers are able to sense the acceleration in different directions. Therefore, the sensed values must be linked with the corresponding measured variable and unit in order to avoid interpretation errors.

Apart from the information identified in the different datasets, we believe that a common information system should also implement the following functionalities for fulfilling the mentioned challenges:

- *Data standardization*: store data in a standardized format so that information taken from different sources is saved homogeneously;
- *Batch and online interfaces*: implement communication interfaces so that data can be uploaded, integrated to existing data, and downloaded from the main database;
- *Flexible search*: provide a Graphical User Interface (GUI) to facilitate the search over the data stored in the repository.

3. Illustrative examples

Next, the need of a common information system for sharing maintenance data in industry and academia is shown with two illustrative examples.

3.1. Industrial example

A multinational has two headquarters, each of them has similar production units (Fig. 1). The unit consists of one three-phase induction motor connected to a shaft lifted with two ball bearings. The shaft moves a conveyor belt for the transport of products. A frequent failure within the unit occurs in the inner raceway of the bearings. The multinational decides to perform PHM analyses in order to improve the system performance by means of better maintenance practices. Maintenance engineers know that failures in the inner raceway of the bearings increment the overall system vibration and unbalance the symmetrical three-phased currents of the motor's stator. Therefore, operators are asked to install current sensors on the units of the headquarter 1, while acceleration sensors at headquarter 2. Each of the headquarters collects operational data from the sensors in an independent manner and stores them in files. A single global engineer from the multinational receives and analyzes this information in order to generate a prognostic model of the unit for inferring future performance. The integration of the data from the two plants provides more inputs for the model development enhancing the quality of the prediction.

3.2. Academia example

The Politecnico di Milano's School of Management owns the Industry 4.0 research laboratory shown in Fig. 2. This laboratory has a prototype production line for assembling the components of a mobile phone. The line consists of seven workstations. Among other features, this laboratory allows researchers to monitor the asset state by means of sensors placed in each workstation. Data from these sensors can be used to perform PHM analysis concerning the whole manufacturing line and each workstation.

The Universidad de los Andes and the Politecnico di Milano want to start collaborating in a PHM-related project. After a first analysis, the prototype production line of the Politecnico di Milano is selected as case study. Now, the Politecnico di Milano has the need to share maintenance data with the Universidad de los Andes.

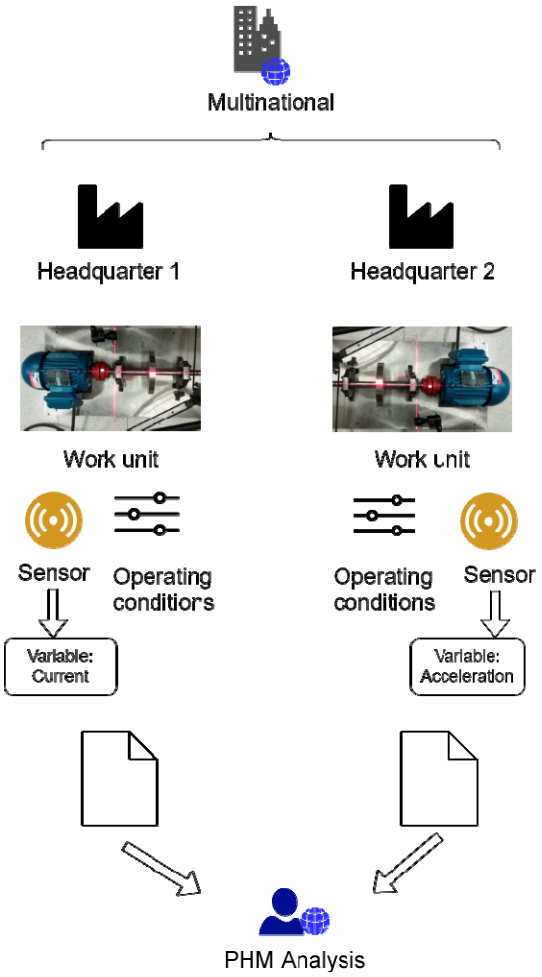

Fig. 1 Illustration of a possible industrial scenario.

Fig. 2. Industry 4.0 Laboratory at the Politecnico di Milano.

4. XRepo

XRepo stands for eXperiments REPOsitory. It is an information system able to collect, standardize and store experimental data, and provides functionalities to categorize, search and generate data reports. Data can be uploaded by files (batch) or via web services (online). This section explains the developed XRepo and is organized as follow: a domain model able to represent the maintenance context is illustrated in section 4.1. Section 4.2 defines a standard format for uploading sample data to the repository, while section 4.3 presents an architecture able to fulfill the functionalities identified in the final part of section 2. Section 4.4 illustrates a GUI that allows user to utilize the XRepo functionalities. Eventually, the selected technologies for the deployment of XRepo are reported in section 4.5.

4.1. Domain model

We studied related work, publicly available datasets, and datasets provided from the two research groups involved in the study. As a result, we came out with a domain model that abstracts the entities related to the domain (Fig. 3). Next, each entity is illustrated:

- *Organization*: the root concept of the model. The organization groups together the information loaded in the repository by a particular group of people, e.g., the multinational, headquarters, research groups, etc.
- *Target System*: represents a specific system that can be under observation and operates under a specific set of conditions; e.g. a production line, a motor, etc. The system has a set of operative ranges which identify specific variables to which the system is constrained; e.g. velocity [0rpm,2000rpm]. In the illustrative example of section 3.1, the work unit consists in an instantiation of the target system.
- *Experiment*: works as a technical sheet for a window of observation that the user wants to execute on the target system. The sheet can hold any kind of information related with the observation as for example objective, methodology, conclusions, execution under normal conditions, attempt of a bearing failure, etc.
- *Sampling*: represents the action of taking samples from the target system when it runs under a specific set of operative conditions (e.g. velocity of 1500rpm), using a specific layout of sensors and during a period of time. Users must define operative conditions that are within the operative ranges defined at the target system level. Devices and sensors are added to the sampling for the purpose of describing a complete technical sheet of the sampling process. Sensors are elements that scan specific aspects of the target system. In turn, Devices collect the data from one or more sensors. In the illustrative example of section 3.1, the sensors are current and acceleration, while the device is the acquisition card to which each sensor is connected.
- *Sample*: represents a snapshot of a sensor on a specific instant of time. This snapshot is a measure of the sensed physical variable. Each sample is associated with a sampling and a sensor.
- *Tag*: entity that can be linked to experiments or sampling. Tags work as metadata to facilitate the search process and organize the information. Searching by criteria is important since specific data must be selected based on the purpose of the PHM analysis. For example, both faulty and normal condition must be used for training and validating supervised learning models.

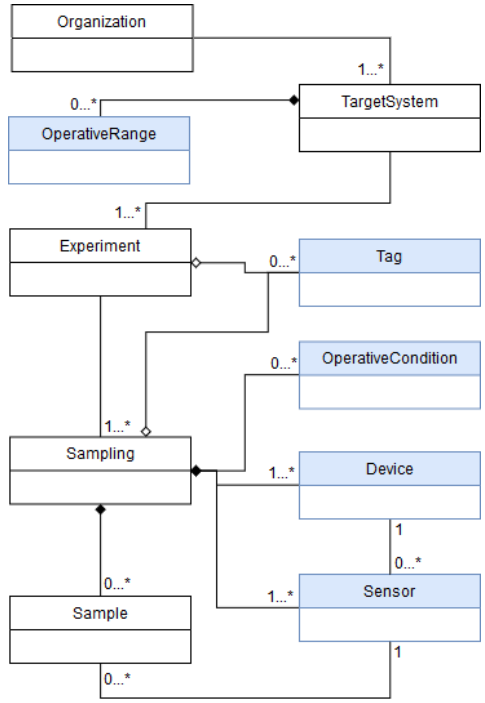

Fig. 3 Domain model.

4.2. Standard Format for Samples

Samples measured from a manufacturing system are the key information entered to XRepo. Samples are the inputs needed to assess the system health and to predict future performance. The definition of a common format for data exchange is fundamental since samples need to be imported and exported to/from the repository.

The following data exchange format is proposed:

- *sampling_id*: a unique id of a Sampling record already created in the instantiation of the domain model;
- *date_time*: represents the local date time when the sample was measured, in yyyy-MM-dd'T'HH:mm:ss.SSS format. This information is valuable when users are looking for data from an experiment that is run in a local time range;
- *timestamp*: represents the time instant based on the sensor or device clock. This information is used to sequentially organize the samples;
- *sensor_id*: represents the identification of a sensor used in a particular sampling; e.g. accelerometer1;
- *measure_variable*: name of the variable scanned from the sensor; e.g. acceleration_x;
- *variable_value*: value for the scanned variable; e.g. 0.0160. Information of the corresponding unit can be identified on the sensor metadata of the domain model.

If a sensor scans multiple variables, each record will be represented with the following pairs: [*measure_variable*, *variable_value*]. In XRepo, two possibilities were implemented for uploading samples represented with the aforementioned standard format: i) CSV file with the standard format fields separated by comma; ii) web service that receives the standard format fields formatted as JSON. It can be noticed that XRepo constrains samples to be represented through the identified standard format. An example of implementation of the proposed standard format is shown in Listing 1.

Listing 1: Example of the XRepo standard format fields

```
{
  "samplingId": "5ce611b86334053ef0947233",
  "sensorInternalId": "accelerometer_1",
  "dateTime": "2019-05-25T23:58:06.470Z",
  "timeStamp": {
    "epochSeconds": 1558812004,
    "nanos": 463000000
  },
  "measurements": {
    "acceleration_x":0.154894,
    "acceleration_y":0.074463
  }
}
```

4.3. Software Architecture

XRepo has been implemented as a web application and stores the datasets in a MongoDB database. The web application allows to fulfill the following functionalities: i) create the scaffolding identified in the domain model; ii) upload sample data represented in the defined standard format; iii) download data that satisfies search criteria. Fig. 4 shows the architecture components of the application.

- *Management components*: this category includes the components named Users, Target System and Experiment. The three components share the Create, Retrieve, Update, Delete functionality to instantiate the entities identified in the domain model. In general, they allow to parameterize the experiment whose samples will be stored in the repository.
- *Samples functionalities*: this category groups two functionalities: Search Samples and Load Samples. The former allows users to find samples stored in the repository based on user criteria. The latter allows users to load into the repository samples associated to an existing sampling by using one of the possible file formats; i.e. CSV and JSON.
- *Processors of background tasks*: correspond to the Data Exporter and File Processor components. These two components implement time and resource consuming operations that run as background tasks according to the infrastructure capacity. The Data Exporter creates files containing samples that match search criteria. The File Processor reads and stores samples that were loaded via the standard format files.

In addition to the previously mentioned components, XRepo has a GUI component named *Web UI* to interact with the user. Eventually, a *Security* component is implemented to authenticate and authorize access to the application resources.

It is important to note that the component *Experiment Data Adapters* is not part of XRepo and must be developed by each stakeholder in order to convert sample files that conform to a source format into files conforming to XRepo standard format. This translation guarantees that the data can be straightforwardly uploaded to XRepo.

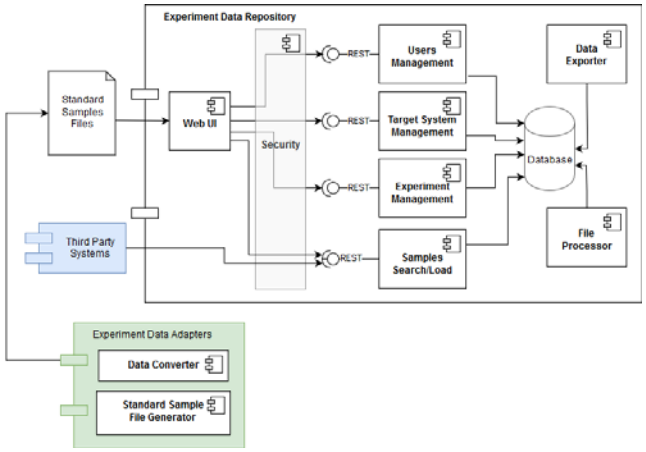

Fig. 4 Components diagram.

The deployment of the identified architecture is shown in Fig. 5. The system has four main artifacts distributed in three different servers. The distribution of the artifacts across the servers had as rationale to divide the system operational load and to separate concerns.

The *Web Server* contains the front artifact to interact with the users. It includes all the management components, the GUI and the samples functionalities.

The *Database Server* contains a running instance of MongoDB to store all the repository information, and the *Worker* component. This component is in charge of executing the background tasks: i) generation of reports resulting from a search; ii) processing of sample files loaded through the GUI. The Worker and the Database were located in the same server to increase the performance, since the background tasks rely on the database to insert or read data.

Eventually, the *File Server* is the repository in which all the generated reports and the files pending to be loaded into the database are stored. The access to this server is through a Network File System mounted on both the Database and the Web Server.

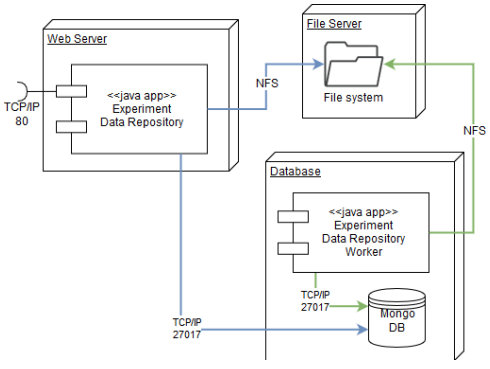

Fig. 5 Deployment diagram.

4.4. Graphical User Interface

This section presents the GUI that allows the user to utilize the XRepo functionalities.

Loading Samples (Fig. 6): this functionality allows to upload samples to the repository using a CSV file with records in compliance with the defined standard format. After the file update from the user, a pending task is created. The task will be processed when a periodic executed process detects the pending task and enough threads are available to process the task.

Search Samples (Fig. 7): provides a template in which search criteria are entered to search within the repository for samples with the selected characteristics. A search requires a target system, while all other fields are left as optional. The user can search from samples that were measured between a date range, marked with a set of tags, associated with specific sensors, and whose experiment was executed under specific operative conditions. As occurred for the load functionality, a pending task is created in the application and its execution is later addressed.

Batch Tasks: this functionality provides insights about the tasks requested from the user. Here, the user is able to check in real time the progress of each task and can download the generated files from the sample search tasks. Examples for the load sample and report generation progresses are respectively reported in Fig. 8 and Fig. 9.

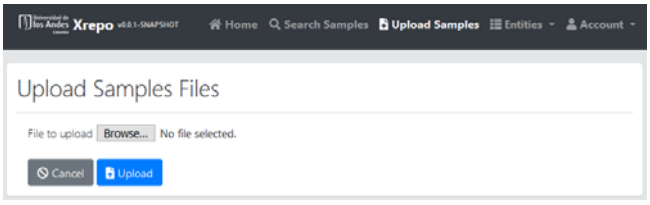

Fig. 6 Upload samples form.

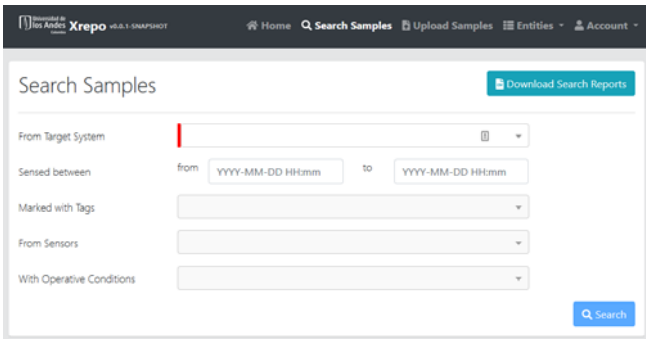

Fig. 7 Search form.

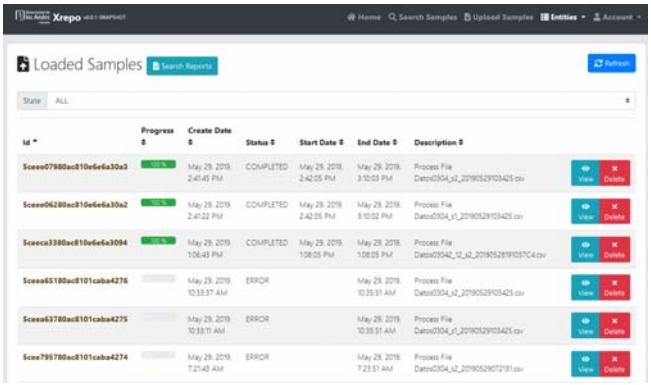

Fig. 8 Load samples progress.

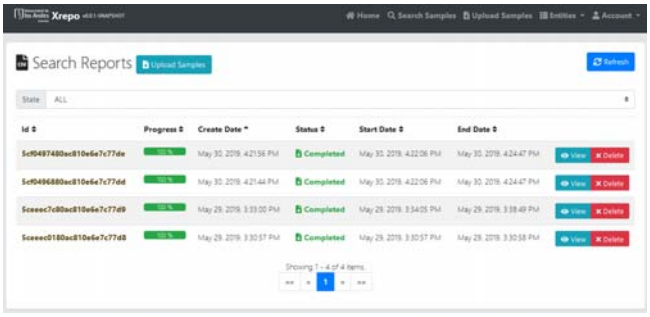

Fig. 9 Report generation process.

4.5. Technologies underlying XRepo

XRepo was built through the JHipster framework [19]. JHipster provides an integrated stack of technologies to create an agile development environment. XRepo uses Spring Boot as backend framework deployed on an embedded Tomcat 8 server, while Angular 7 is used for the frontend GUI.

MongoDB was selected as database. With respect to other databases as NoSQL, MongoDB has the following advantages: i) offers a flexible schema to store the information of the samples; ii) presents better performance for the upload and simple queries [20]–[22].

5. Validation

A preliminary validation was designed for testing the functionalities of XRepo and for verifying the integrity of the data uploaded in the repository.

5.1. Batch Upload

The batch validation was carried out from the Grupo de Investigación en Automatización para la Producción (GIAP) of the Universidad de los Andes. This research group has an experimental test bench to study failures in ball bearings by analyzing current and acceleration data. The experimental test bench consists in the production unit described in section 3.1.

The GIAP group accessed XRepo and instantiated the domain model with the information of their test bench. They also included tags for labeling the different experiments and samplings. After the creation of an adapter to convert the sample files to a CSV file that implements the standard file format of XRepo, they were able to upload the sample files obtained from their experiments. Then, they selected specific data through the search functionality and downloaded it from XRepo. Eventually, the downloaded data was processed.

In the experiments of the GIAP group, current and acceleration were acquired with two different devices. Therefore, samples were saved in different formats and a different code had to be implemented in their analysis for the reading of the data. Whereas, the samples downloaded from XRepo were represented with a standard format. Therefore, the same code was used for data reading independently from the type of data; i.e. current or acceleration. This was identified from the GIAP group as a benefit of using XRepo.

5.2. Online Upload

Online validation was carried out by the Politecnico di Milano, School of Management, Manufacturing Group research team. The research team had data related to the monitoring of a drilling machine in their Industry4.0 laboratory. Data were collected by an accelerometer connected to a Raspberry-pi, whose task was to gather the vibration on the 3 axes (x, y, z) of the drilling device inside the station. These data were enriched with a timestamp and a more readable data field, plus a tag indicating normal or faulty samples. This data was saved as JSON files.

The research group used the XRepo GUI for instantiating the domain model for their sampling, and then developed a Javascript application for formatting their data with the XRepo standard format. Then, via the REST service provided by the developed solution, their newly formatted JSON data were sent and uploaded to the platform.

Then, it was possible to request data that presented the 'faulty' tag, download them form the repository and analyze them.

5.3. Data Integrity

Within their analysis, the GIAP group calculated the Fourier transform of current samples at different percentage of useful life of a rolling bearing (Fig. 10). This analysis was performed on samples obtained directly from the acquisition device and from the same samples uploaded to and then downloaded from XRepo. The exact same plot was obtained showing that XRepo does not modify the information content of the samples.

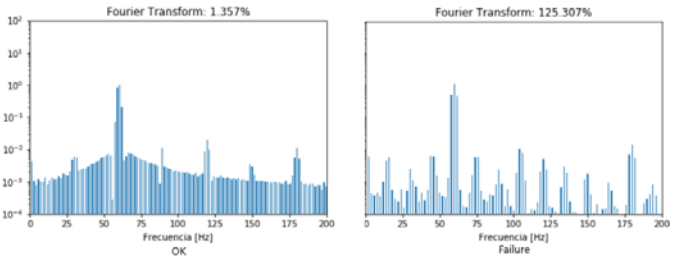

Fig. 10 Fourier transform of the motor's stator current at different percentages of the bearing useful life.

6. Conclusions and future work

In this paper, XRepo was proposed as a common information system for maintenance datasets. XRepo allows to upload / tag / search / download experimental data taken from manufacturing systems. This application enables collaboration among research groups and companies due to its sharing capability. The stored data can be exported and PHM analysis can be performed to provide insights about the manufacturing system behavior and to predict possible failures during operation.

The validation shows that XRepo fulfills the proposed functionalities: i) collects all the metadata and samples associated to experiments without compromising the integrity of the data; ii) allows to upload data both in batch and online;

iii) enables the categorization and searching of data by criteria.

Eventually, possible future work is listed:

- Granular access control over the stored information must be implemented. Currently, any user can modify the information, even if the user is not the owner of the data. XRepo should restrict the modification rights to only the owner, and allow read access to anyone around the world;
- Implement identified functionalities that can improve the usability of the information system. For example, a functionality for converting physical units should be available. In fact, an information can be entered with a certain physical unit, while user may utilize a different one during the search by criteria. In this situation, the search would fail since the information present different units;
- Carry out additional validations in which data from more organizations/research groups will be studied to deeply evaluate the solution, identify additional functionalities, and eventually enrich the data model. For example, the current version of XRepo allows only to store waveform type data; i.e. time series. However, other types of data can be used in PHM analysis such as multi-dimension type; i.e. images;
- Investigate how the data volume challenge can be faced.

Acknowledgments

The authors would like to thank Julian Rodriguez of the GIAP research group for the batch upload and the data integrity validation.

References

[1] A. C. C. Tan, A. Heng, J. Mathew, and S. Zhang, “Rotating machinery prognostics: State of the art, challenges and opportunities,” *Mech. Syst. Signal Process.*, vol. 23, no. 3, pp. 724–739, 2009.

[2] J. Lee, E. Lapira, S. Yang, and A. Kao, “Predictive manufacturing system - Trends of next-generation production systems,” in *IFAC Proceedings Volumes (IFAC-PapersOnline)*, 2013, vol. 46, no. 7, pp. 150–156.

[3] L. Fumagalli, L. Cattaneo, I. Roda, M. Macchi, and M. Rondi, “Data-driven CBM tool for risk-informed decision-making in an electric arc furnace,” *Int. J. Adv. Manuf. Technol.*, 2019.

[4] H. Lasi, P. Fettke, H.-G. Kemper, T. Feld, and M. Hoffmann, “Industry 4.0,” *Bus. Inf. Syst. Eng.*, vol. 6, no. 4, pp. 239–242, 2014.

[5] J. Lee, B. Bagheri, and H. A. Kao, “A Cyber-Physical Systems architecture for Industry 4.0-based manufacturing systems,” *Manuf. Lett.*, vol. 3, no. October 2017, pp. 18–23, 2015.

[6] H. Kagermann, W. Wahlster, and J. Helbig, *Securing the future of German manufacturing industry: Recommendations for implementing the strategic initiative INDUSTRIE 4.0*, no. April. Forschungsunion, 2013.

[7] K. Englmeier, “Role and importance of semantic search in big data governance,” in *Big-Data Analytics and Cloud Computing: Theory, Algorithms and Applications*, M. Trovati, R. Hill, A. Anjum, S. Y. Zhu, and L. Liu, Eds. Cham: Springer International Publishing, 2016, pp. 21–35.

[8] J. Lee and E. Lapire, “Recent Advances and Trends in Predictive Manufacturing in Industry 4.0 Environment,” *Reliab. A Cult. Reliab.*, vol. 1, no. 1, pp. 38–41, 2016.

[9] J. Lee, H. Qiu, G. Yu, J. Lin, and Rexnord Technical Services, “Bearing Data Set,” *NASA Ames Prognostics Data Repository*. IMS, University of Cincinnati, Moffett Field CA, 2007.

[10] K. A. Loparo, “Case Western Reserve University Bearing Data Center,” *Bearings Vibration Data Sets, Case Western Reserve University:http://csegroups.case.edu/bearingdatacenter/home*, vol. www.eecs.c. pp. 22–28, 2012.

[11] P. Nectoux *et al.*, “PRONOSTIA: An experimental platform for bearings accelerated degradation tests.,” in *IEEE International Conference on Prognostics and Health Management, PHM’12*, 2012, pp. 1–8.

[12] I. Roda, M. Macchi, and L. Fumagalli, “The Future of Maintenance Within Industry 4.0: An Empirical Research in Manufacturing,” in *Advances in Production Management Systems. Smart Manufacturing for Industry 4.0*, 2018, pp. 39–46.

[13] Penn Sate University Applied Research Laboratory, The Boeing Company, and MIMOSA Foundation, “Open System Architecture for Condition-based Maintenance (OSA-CBM),” no. August. 2006.

[14] M. A. L. Campos, L. Fumagalli, J. F. G. Fernandez, A. C. Marquez, and M. MacChi, “UML model for integration between RCM and CBM in an e-Maintenance architecture BT,” in *1st IFAC Workshop, A-MEST’10, Advanced Maintenance Engineering, Services and Technology.*, 2010, vol. 1, no. PART 1, pp. 110–115.

[15] A. Vijayaraghavan, W. Sobel, A. Fox, D. Dornfeld, and P. Warndorf, “Improving Machine Tool Interoperability Using Standardized Interface Protocols: MT Connect,” *2008 Int. Symp. Flex. Autom.*, pp. 21–24, 2008.

[16] M. Vieira, N. Mendes, J. Durães, and H. Madeira, “The AMBER Data Repository,” *Work. Resil. Assess. Dependability Benchmarking*, 2008.

[17] M. Lichman, “UCI Machine Learning Repository [http://archive.ics.uci.edu/ml].,” *UCI Machine Learning Repository*. p. 2013, 2013.

[18] A. Von Birgelen, D. Buratti, J. Mager, and O. Niggemann, “Self-Organizing Maps for Anomaly Localization and Predictive Maintenance in Cyber-Physical Production Systems,” in *Procedia CIRP*, 2018, vol. 72, pp. 480–485.

[19] J. Dubois, “Generate your Spring Boot + Angular/React applications.” 2013.

[20] Z. Parker, S. Poe, and S. V. Vrbsky, “Comparing NoSQL MongoDB to an SQL DB,” in *Proceedings of the 51st ACM Southeast Conference on - ACMSE ’13*, 2013, p. 1.

[21] M. Stonebraker, “SQL databases v. NoSQL databases,” *Commun. ACM*, vol. 53, no. 4, p. 10, 2010.

[22] J. S. Van Der Veen, B. Van Der Waaij, and R. J. Meijer, “Sensor data storage performance: SQL or NoSQL, physical or virtual,” in *Proceedings - 2012 IEEE 5th International Conference on Cloud Computing, CLOUD 2012*, 2012, pp. 431–438.