A solution for annotating sensor data streams - An industrial use case in building management system

Dumindu Madithiyagasthenna*, Prem Prakash Jayaraman*, Ahsan Morshed†, Abdur Rahim Mohammad Forkan*, Dimitrios Georgakopoulos*, Yong-Bin Kang*, Mirek Piechowski ‡
*Department of Computer Science and Software Engineering, Swinburne University of Technology, Melbourne, Australia
†Central Queensland University, Melbourne, Australia
‡Piechowski Energy, Melbourne, Australia
dumindudm@gmail.com

***Abstract*—Smart buildings equipped with various building management systems and digital control systems produce enormous amounts of sensor data that can be used to investigate and diagnose operational issues such as unsatisfactory thermal comfort outcomes, excessive energy consumption and/or predicting failures before they occur. However, current building management systems often face the issues with incomplete or unstructured metadata associated with sensor data which prevent such pro-active, predictive and prescriptive analysis. Currently, building service engineers manually map the sensor data streams to aid their diagnostic process. This process is expensive, ineffective and is also prone to human errors. This paper proposes a novel semi-automated approach that annotates incoming sensor data streams. We also propose extensions to Project Haystack, a well-known ontology used for naming conventions and taxonomies for building equipment and operational data. We have developed a tool that is currently used by our industry partner and incorporates the proposed automatic annotation approach and maps the data streams to our Haystack-extended ontology. The tool includes an easy to use interface for engineers to easily diagnose issues in mechanical building services. The proposed approach has been validated via both usability and technical evaluation.**

***Index Terms*—Smart Building, Sensors, Ontology, IoT**

I. Introduction

Smart buildings include mechanical services such as heating, cooling, ventilation, vertical transportation etc., that are equipped with sensors that produce data about the service. The sensors and the mechanical services are mainly controlled by digital control systems that are integrated by a master controller which provides access to slave controllers' data and data storage.

In the commercial, or non-industrial, sectors these integrated control systems are usually referred to as Building Management Systems (BMS). Almost every building's operational data generated by BMS uses a unique non-standard naming convention to tag the sensor data stream with metadata [1]. For example, it is very common that the same variable, "Outside Air Temperature", is tagged with different names by different BMS. Adding to this, there is no consensus of a single data standard for BMS data streams and this creates significant data heterogeneity problems. This introduces the issue of incomplete or unstructured metadata associated with data streams and represents a significant obstacle for automated data analysis at scale [2], [3]. Furthermore, the metadata available from the BMS environment (and the underlying sensors) are of low-quality and not well described [4].

Building service engineers called to investigate and diagnose operational issues of mechanical services. This diagnostic process usually consists of at least three diagnostic steps; the analysis of the physical configuration of a mechanical system, retrieval of historical sensor data from BMS and the analysis of BMS historical data. Engineers who perform these tasks to extract useful information out of these data streams, generally mechanical engineers, need to have very high domain expertise and knowledge of the unique naming convention of the particular BMS to interpret and analyse the data streams [5]. Typically, the process of BMS data analysis [6] is time-consuming and involves several preparatory steps as outlined earlier, before the data can be analysed.

In its basic form, data analysis involves graphing relevant variables and visual analysis by an experienced engineer. Most, if not all, of those diagnostic tasks, can be performed automatically and at scale taking advantage of appropriately formulated algorithms. Besides, building data can provide a rich source of insight into building operation far exceeding a mere fault diagnostic. These insights can vastly improve both building performance and increase the productivity of building management operations. Consequently, if implemented at scale, they can drastically improve the profitability of commercial real estate (CRE), and other, asset portfolios.

To address the aforementioned issue that the industry currently faces in automatically interpreting sensor data streams from BMS that use different naming conventions, in this paper, we propose an ontology that can hold more rich and contextual meta-data, built on top of the existing Project Haystack ontology [7], and a novel algorithm,

2375-0324/20/$31.00 ©2020 IEEE
DOI 10.1109/MDM48529.2020.00042

namely Semi-Automated Technique for Annotating Sensor data streams (AGSDA), that uses hierarchical clustering to semi-automatically map BMS metadata following different naming convention on to the proposed extended Haystack ontology. We also implement a tool that incorporates AGSDA for engineers to facilitate fast analysis of sensor data streams for various purposes such as diagnostics, analysis and predicting failures. The quality of the hierarchical clustering has been evaluated using a well-known clustering evaluation metric and the usability of the tool has been evaluated via a usability study that included both expert and non-expert users. In particular, this paper makes the following contributions:

1) An ontological model that extends Project haystack to provide more contextual information of sensors with their relationships to each other;
2) AGSDA, an algorithm which can semi-automatically map a sensor data streams from a BMS to haystack ontology
3) A tool that incorporates AGSDA that provides engineers with an easy interface to explore and analyse sensor data streams originating from BMSs. The developed tool is currently being used by our industry partner *Piechowski Energy* as part of their building management product offering.
4) Experimental evaluations validating the accuracy of AGSDA and a usability study to evaluate the usefulness of the proposed tool that incorporates AGSDA

The rest of the paper is structured as follows; Section II describes surrounding work relating to this topic, Section III describes the proposed Ontology, Section IV illustrates the developed algorithm for mapping sensor data; Section V describes the proof of concept developed; Section VI discusses the evaluations of the proposed system and finally Section VII concludes the paper.

II. Related Work

There have been various attempts at extracting metadata from building systems sensor data. Some use techniques such as Random Forrest to train a classifier which can predict the type of sensor data by learning statistical features embedded in the numerical data such as mean, max, variation, etc. [5], [8], whose accuracy is very sensitive to environmental factors such as weather [5]. These attempts have been directed at classifying the types of sensor point types [9], (for example if a particular data point is of the type temperature, or pressure, or volume). But this fails to extract the semantic and contextual information encoded within these sensors. There have been more linguistic approaches which have tried to extract information from the sensor tag names. Some of these efforts have been focused on using linguistic similarity values of sensor sub-tags to match them against ontologies like Project Haystack semi-automatically [10]. But simply using the highest similarity value to match a sub-tag name only yielded an accuracy of 16%. The Zodiac system [11] combines numerical data and linguistic information using hierarchical clustering and an active learning random forest to reduce the workload on the expert in semi-automatic approaches. Another prominent system introduced the combining of all these linguistic methods to extract the meta-data encoded in the sensor tags. They used a programming-by-examples approach with hierarchical clustering and string similarity along with a sub-string extraction language to map meta-data into Project Haystack tags [9]. Our attempt is an extension of this system with a focus on refining the clustering quality and mapping it to a general ontology with more contextual and relationship information.

In the past ten years, or so, there has been an effort by private and industry organisations to develop an automated process for building sensor data analytics. Project Haystack [7] is an example of industry-led, international effort focused on developing semantic modelling solutions and a common vocabulary for data from buildings control systems. However, it is a flat ontology, with minimal hierarchical structure and standard relationships. There exist other metadata standards like Industry Foundation classes standard (IFC) used in construction that aims to provide a unified format for BMS [12]. However, it is highly complex and requires a lot of information for each tag, and since data can be missing and constantly changing even within similar sensor points, it is impractical to use IFC. Semantic Sensor Network (SSN) [13], another initiative which provides an ontology for sensor observations but its high complexity is more suited for developers of BMS systems. Our ontology aims at bridging this gap by developing a schema which can be adapted to any BMS system with little understanding of the underlying details, while also providing rich hierarchical information of the sensors and the system as a whole.

III. Extended Haystack Ontology

Many building systems tend to encode all of the metadata relating to a sensor such as location, type, equipment in one string – its "sensor tag" [9]. For example, the tag *FC 1_Outside Air Temp* encapsulates the sensor's type (Temp), the equipment it is a part of (FC), and attributes relating to it (Outside Air). There is no schema or ontology available that describes these links and hierarchies.

Project Haystack is an attempt at standardising individual sub-tags, therefore we have used an adapted version of their taxonomy to develop an ontology along with an expert. This ontology aims to describe sensor metadata in a more intuitive way for the experts looking at it, and also be machine-processable at the same time.

The ontology contains a hierarchy which explains where a sensor is, which equipment it is a part of, what are its components and attributes, and what type it is. It also contains a reference to each of these values. We call these "types". The high-level hierarchy of the types is shown

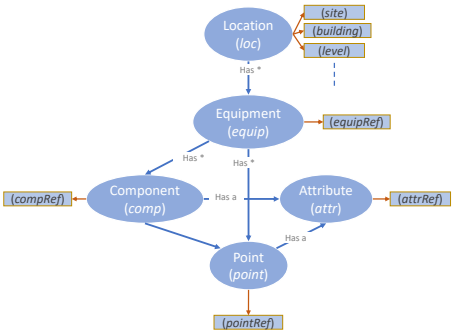

Fig. 1. High-level hierarchy of the ontology

in figure 1, and a breakdown of these types with relating haystack tags is shown in table I.

TABLE I
THE HIERARCHY ALONG WITH RELATING HAYSTACK TAGS

Type	Values
loc	site, building, level, zone
equip	fan coil, heating hot water, chiller, air handling unit, ...
ref	equipRef, compRef, pointRef, ...
comp	water, air, damper, ...
attr	entering, leaving, supply, ...
point	temp, volume, pressure, co2, ...

Most approaches before have looked at point type classification, such a classifying which "point" a sensor falls into (eg: temperature, volume, etc.). Our ontology gives the expert a more holistic view of a sensor and how it relates to other sensors and allows computers to run analysis on it based on any of the types we have defined.

When applied to the sensor tag *ArtsCentre_FC 1_Outside Air Temp*, this would result in

```
{
    "loc": {
        "building": "ArtsCentre"
    },
    "attr":{
        "type":"outside"
    },
    "comp":{
        "type":"air"
    },
    "data_point":"FC 1_Outside Air Temp",
    "equip":{
        "ref":"1",
        "type":"fan coil"
    },
    "point":{
        "type":" temp "
    }
}
```

Listing 1. The ontological representaion of the sensor tag *ArtsCentre_FC 1_Outside Air Temp*.

When stored in a document database, operations on the sensors become very easy to manipulate and execute. For instance, if the expert wants to run analysis on all the pressure points in the 3rd fan coil equipment in the gymnasium building, they can access all of these points simply by using {*loc.building = 'gymnasium' AND equip = 'fan coil' AND ref.equipRef =3 AND point = 'pressure'* }. This would return only the required points.

The advantage of this methodology is apparent since it allows for very complex querying of data points using very intuitive syntax, whereas traditionally, spreadsheets and intricate regular expressions would be needed to filter out required points and is usually a labour extensive task. It should be known that some vendors leave certain metadata out of the sensor tag so it is impossible to infer these values from the tag alone. Table II shows two subsets of sensor tags, obtained from 2 different vendors.

For instance, Group 1 has no location information but has very rich equipment, component and attribute information. Group 1 has been obtained from one building, therefore the location information such as site and building can be applied manually for the group as a whole.

Group 2 has very extensive location information but minimal information about equipment and attributes. This group is from a BMS that is controlling multiple buildings within a site therefore location information is crucial. It is almost impossible to infer any equipment, component and attribute data from this group and is impractical for an expert to tag all that information individually. However, if it is known which buildings/levels are controlled by certain equipment, this equipment information can be applied directly to the sensors with the appropriate location data.

Table III is an example of the result of mapping a sensor out of each of the two groups into our described ontology, by inferring the maximum information available from the tags using our algorithm.

IV. AGSDA: Semi-Automated technique for annotating sensor data streams

Our algorithm, AGSDA (AnnotatinG Sensor Data streAms) utilises the programming-by-examples methodology [14], which builds a set of programmatic rules and repeats what the user has done following some examples.

As shown in table II, the sensors within a group/vendor have similar syntactical schema but this schema can vary semantically within a group. Therefore, using the programming-by-examples method for the whole dataset would not work since the regular expression model it will build would need to be too complex to cater to all the different tags. Hence, we use an adapted version of the algorithm that is discussed in [9].

Algorithm 1 contains several steps, described below. The first being **feature extraction**; this converts the set of raw textual sensor tags into a symbolic representation since the tags can have very distinct semantic relationships. Let $\{R_1 ... R_n\}$ be a list of raw sensor tags. We initially strip the prefix common to the whole data set from the tags which allow us to detect more subtle similarities between tags (this is especially true for Group 2 in table II, where the site name makes up a significant portion of the tag name). Symbolic features are extracted by converting alphabetic characters to the letter "1", numeric to "2", white-spaces to "3" and all other characters to "4". Adjacent alphabetic and numeric characters ("21" and "12") are also made to be "2" to give them the same

TABLE II
SAMPLE OF TWO DIFFERENT SETS OF SENSOR TAGS FROM DIFFERENT VENDORS

Group1	Group2
• Chiller 1_CHW 1 Entering Temp • Chiller 1_CHW 2 Leaving Temp • Chiller 2_CHW 2 Entering Temp • Chiller 2_CHW 2 Leaving Temp • Chillers_Common Header Temp • Chillers_Field Flow Temp • Chillers_Field Return Temp • FC 1_Exhaust Air Temp • FC 1_Exhaust Air Volume • FC 1_Outside Air Temp • FC 1_Return Air Temp	• MELBOURNE SCHOOL / ARTS CENTRE / AIR CON / LEVEL 1 / W123 Office / ZONE TEMPERATURE • MELBOURNE SCHOOL / ARTS CENTRE / AIR CON / LEVEL 1 / W117 Latin Classroom / ZONE TEMPERATURE • MELBOURNE SCHOOL / GYMNASIUM / AIR CON / LEVEL 1 / ALARM MONITOR • MELBOURNE SCHOOL / ARTS CENTRE / AIR CON / LEVEL 2 / W225 Artroom Ducted / ZONE TEMPERATURE • MELBOURNE SCHOOL / ARTS CENTRE / AIR CON / LEVEL 2 / W227 Science Lab A / ZONE TEMPERATURE • MELBOURNE SCHOOL / GYMNASIUM / AIR CON / LEVEL 2 / W217 Office / ZONE TEMPERATURE

TABLE III
EXAMPLE OF MAPPING SENSOR TAGS TO THE ONTOLOGY

Group1

```
{
    "data_point": "FC 1_Exhaust Air Temp",
    "location": "",
    "equip": {
    "type": "Fan Cooler",
        "ref": "1"
    },
    "comp": {
        "type": "Exhaust",
        "ref": ""
    },
    "attr": "Air",
    "point": "Temp"
}
```

Group2

```
{
    "data_point": "MELBOURNE SCHOOL / GYMNASIUM / FIRST FLOOR / AC
        ↪ −1−06 Yr 6 Staff Office / ZONE TEMPERATURE",
    "location": {
        "site": "CAMBERWELL GRAMMAR",
        "building": "MIDDLE SCHOOL",
        "level": "FIRST FLOOR",
        "zone": "AC−1−06 Yr 6 Staff Office"
    },
    "equip": "",
    "comp": "",
    "attr": "",
    "point": "Temp"
}
```

weight as numbers since reference systems in most BMS use a combination of these (eg: 2A, 3C, etc.) to number components in the building. Finally, these feature vectors have their repeating features merged, and their lengths normalised by padding them with "0", giving us a list of normalised feature vectors $\{R_1^{V_n}...R_2^{V_n}\}$. An example of this process is showed in figure 2

Agglomerative (bottom-up) hierarchical **clustering** is performed on $\{R_1^{V_n}...R_2^{V_n}\}$ using the UPGMA algorithm [15] by the Clust function, which uses an unweighted mean to form flat clusters, with a set threshold of 0. This process is described in detail later on. The result is a set of clusters $\{C_1...C_n\}$. An example of this process is displayed in figure 2.

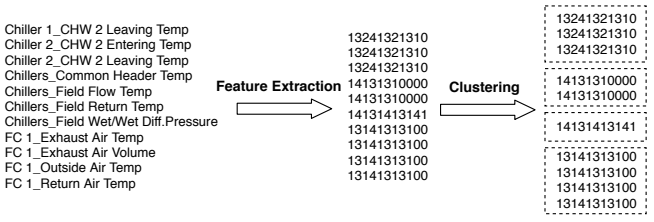

Fig. 2. An example of the process of feature extraction and clustering

One randomly selected sensor tag from each cluster is presented for **expert tagging** (*ExTag*), where the expert needs to tag or "answer" these examples (denoted by A) where $A_i < T^S{}_i, O^T{}_i, O^V{}_i >$ are the expert provided sub-tags, their types and values (from our described ontology O) respectively. This information can be entered through the user-interface we have developed figure 3. Figure 4 shows how an expert might answer and how it might be represented in our ontology.

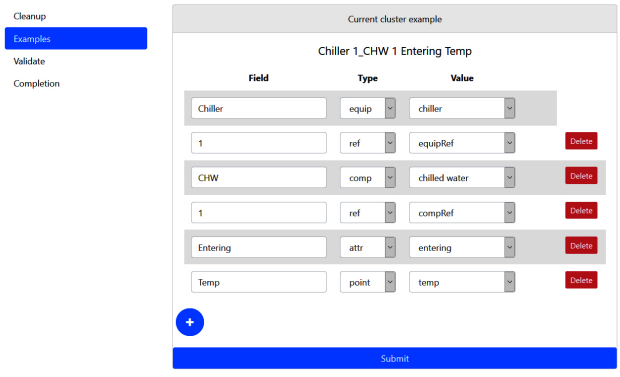

Fig. 3. The interface used for expert tagging in our system

The 4th step is **string splitting** where, based on the $A_i < T^s{}_i >$ and their positions for each cluster, the rest of the tags in the cluster are split accordingly. This is per-

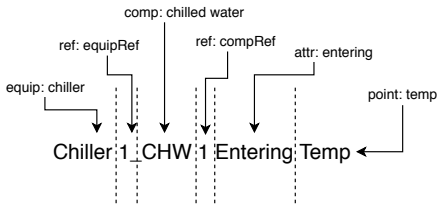

Fig. 4. How expert tagging is interpreted within the ontology

formed by the *FlashFill* function [14] which automatically uses the given user examples to develop rules to apply to the rest of the data, to return $\{C_1 < \{T^s{}_i ... T^s{}_i\} > ... C_n < \{T^s{}_i ... T^s{}_i\} >\}$, a list of sub-tags for each cluster (C).

The next steps finalise the **mapping**. The types from the answers, $A_j < O^T{}_j >$, are assigned to all the types of the sub-tags $C_i < T^s{}_i >^T$, at those positions. Every sub-tag that isn't of type "ref" is **string matched** using the values in our ontology O^V, for that specific type $O^V[A_j < O^T{}_j >]$, to give a list of string-matched sub-tags values, $\{S_1 ... S_n\}$.

Validate denotes the **expert validation** of these sub-tags to produce $\{S^V{}_1 ... S^V{}_n\}$, a list of verified sub-tags values. Since most of the sensor names in any data set is usually a combination of the same set of words, the number of sub-tags to verify are usually very small and is the only verification step we need. These are stored in the database since they can be used for subsequent mapping of data to reduce the need for expert validation. With time, the sub-tags and the effort required for verification will decrease until only BMS specific obscure tags are left.

Finally, *Apply* will apply these verified sub-tag values, to their respective sub-tag. The final output is denoted by $\{T_1 < T^s{}_i, O^T{}_i, O^V{}_i > ... T_n < T^s{}_n, O^T{}_n, O^V{}_n >\}$, a list of tags, which each has its own set of subtags, with their respective types and values from our ontology.

V. BMS Data Analysis Tool - Implementation

In this section, we provide an architecture of the proposed tool that 1) maps BMS sensor data streams using AGSDA algorithm to the extended haystack ontology and 2) provide an easy to use interface for engineers to explore and analyse the sensor data streams produced by the BMS. The tool is currently integrated as part of the product offering of our industry partner *Piechowski Energy*.

A. System Architecture

Figure 5 provides an overview of the BMS data analysis tool. The system has three main layers; Data Layer, Data Mapping layer and Presentation layer. The system takes input from the BMS and processes the raw sensor tags within the Data Mapping layer while also taking input from an expert in parallel to aid with the mapping process where the resulting data is stored, in the Data layer.

We have implemented the AGSDA algorithm and the proposed ontology to develop a working system that when given a list of sensor tags from any BMS, carries out the process of extracting and mapping the meta-data onto our ontology (figure 6). Figure 6 also shows the interface for the data analytics platform that uses the output of this process. It shows a high-level breakdown of the temperature distribution in relation to the setpoints for 2 buildings in the system. The mapped data has allowed us to easily analyse and categorise data into buildings. There is a multitude of practical use-cases this tool enables, one of which is described below.

- **Point specific analysis**: What the ontology enables us to do after the data has been mapped, is allow

Algorithm 1

Input: $\{R_1 ... R_n\}$
Output: $\{T_1 < T^s{}_1, O^T{}_1, O^V{}_1 > ... T_n < T^s{}_1, O^T{}_1, O^V{}_1 >\}$
$pre \leftarrow GetPre(\{R_1 ... R_n\})$
for $\{R_1 ... R_n\}$ **do**
  $R_i^V \leftarrow Trim\ (R_i,\ pre)$
  $R_i^V \leftarrow Replace(([a\text{-}zA\text{-}Z],1),([\backslash d],2),([\backslash s],3),([a\text{-}zA\text{-}Z\backslash d\backslash s],4),\ R_i^V)$
  $R_i^V \leftarrow Replace(([12—21],2),\ R_i^V)$
  $R_i^V \leftarrow Merge(R_i^V)$
end for
$\{R^{V_{n_1}} ... R^{V_{n_2}}\} \leftarrow Norm(\{R^{v_1} ... R^{v_2}\}\ ,\ O)$
$\{C_1 ... C_n\} \leftarrow Clust(\{R^{v_{n_1}} ... R^{v_{n_2}}\},\ UPGMA,\ 0)$
for $\{C_1 ... C_n\}$ **do**
  $A_i < T^S{}_i, O^T{}_i, O^V{}_i > \leftarrow ExTag\ (C_i[rand(len(C_i))])$
end for
for $\{C_1 ... C_n\}$ **do**
  $\{C_1 < \{T^S{}_i \dots T^S{}_i\} > ... C_n < \{T^S{}_i \dots T^S{}_i\} >\} \leftarrow FlashFill(C_i,\ A_i < T^S{}_i >)$
end for
for $A_1 < T_{s_1} > ... A_n < T^S{}_n >$ **do**
  for $\{C_1 < \{T_{s_1} \dots T_{s_1}\} > \dots C_n < \{T^S{}_n \dots T^S{}_n\} >\}$ **do**
    $C_i < T^S{}_i > [pos(A_j < T^S{}_j >)]^T \leftarrow A_j < OT_j >$
    if $A_j < O_{T_j} > \neq O_T$['ref'] **then**
      $C_i[pos(A_j < T^S{}_j >)]^V = SMatch(C_i < T^S{}_i > [pos(A_j < T^S{}_j >)],'levenshtein',O^V[A_j < O_{T_j}>],\ DB.Get(\{S_{V_1} ... S^V{}_n\}))$
      $\{S_1 ... S_n\} += C_i[pos(A_i < T^S{}_i >)]^V$
    else
      $C_i[pos(A_i < T^S{}_i >)]^V \leftarrow A_i < O_{V_i} >$
    end if
  end for
end for
$\{S^V{}_1 ... S^V{}_n\} \leftarrow Validate(\{S_1 ... S_n\})$
$DB.Store(\{S^V{}_1 ... S^V{}_n\})$
$\{T_1 < T^S{}_i, O^T{}_i, O^V{}_i > ... T_n < T^S{}_n, O^T{}_n, O^V{}_n >\} \leftarrow Apply(\{C_1 < \{T^S{}_i \dots T^S{}_i\} > \dots C_n < \{T^S{}_i \dots T^S{}_i\} >\}^V, \{S^V{}_1 ... S^V{}_n\})$

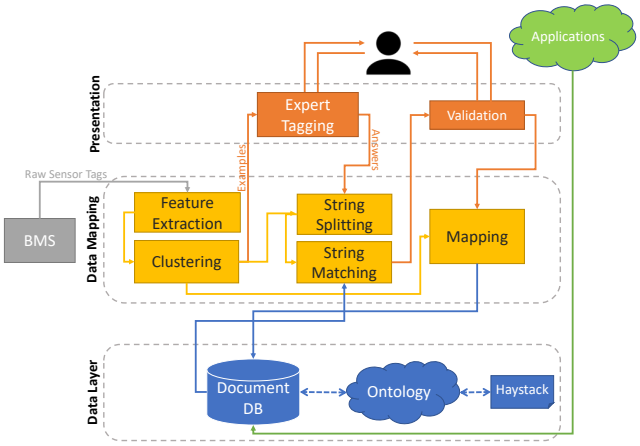

Fig. 5. System Overview

us to query a building to access data with very specific properties. One such use case would be testing if the heaters/cooling systems in a specific building (*loc.building*) are over-heating/cooling. The expert can easily query temperature points (*point.temp*) and analyse their values to a check where in the building (*loc.zone*) are the sensors reporting values outside a set range. This kind of analysis helps in efficient identifying where energy can be saved in a building without the expert having to wrangle through enormous amounts of data. It also removes the need to cater for all different BMS names and schema when querying the data (for example, different BMS can use temp, T, temperature, etc, all to refer to a single type of point, standardised to "temp" in our ontology). Once problems are identified, the mapped sensors also allow the expert to resolve them faster by letting them locate them sooner.

VI. Evaluation

To validate the proposed AGSDA algorithm two data sets were used containing 115 and 112 data points from the different BMS vendors. These data are collected from

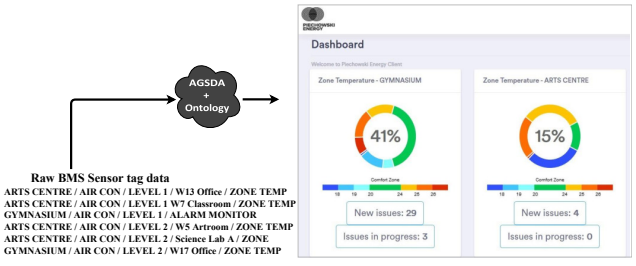

Fig. 6. Overview of how a raw sensor data stream is converted (using our proposed system) into a dashboard (implemented by our industry partner) monitoring temperature point sensors at a building level, displaying a breakdown of the temperature zones, and listing issues which have been identified

a local school in Victoria, Australia. Though our dataset is small, this is a representation of real-world settings. This dataset served our purpose as it is hard to obtain large data for such settings. Only metadata (i.e., tag names) are considered in the evaluation. The system is implemented using python Django and deployed on a cloud server (AWS EC2 instance, 2 GB ram and 3.3 GHz Intel Scalable Processor). Although performance is not the aim of our system it is worth noting that the feature extraction and clustering process was completed in 4 ms and 13 ms for each dataset respectively.

A. Evaluation of AGSDA

Agglomerative clustering, a bottom-up hierarchical clustering approach underpins AGSDS. In such a method the number of clusters does not need to be known prior to clustering which allows us to extract the naturally occurring clusters out of the data set. Each data point starts as their own cluster and then joins based on similarity to form a dendrogram or tree structure figure 7. There exist a plethora of linkage algorithms that can be used for hierarchical clustering. To seek out the most fitting linkage method the Cophenetic Correlation Coefficient (CCC) is used as an evaluation metric. This calculates the goodness of fit and evaluates different types of clustering methodologies.

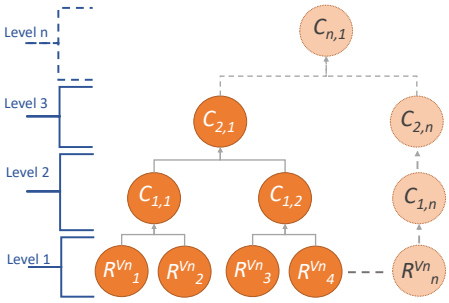

Fig. 7. The agglomerative clustering starting with all points as their own clusters and combining the closest pairs together using UPGMA, until there is only 1 cluster left. Our threshold value helps us obtain the relevant clusters at the right level

CCC was calculated for six common types of linkage methods; ward, median(WPGMC), weighted (WPGMA), complete, single and average (UPGMA), against five popular distance metrics; Euclidean, squared Euclidean (sqeuclidean), Cityblock (Manhattan), Mahalanobis, Minkowski, Hamming, Chebyshev and Jaccard for a group of sensor tags from one building. The CCC values for this are presented in table IV.

We can conclude from this set of evaluations that the average (UPGMA) algorithm performs the best across the board, regardless of the distance metric used. In the average method, the new groups are formed by combining pairs where all pairwise distances contribute equally to each mean, and this is the arithmetic mean distance to all the members within that group.

TABLE IV
THE CCC EVALUATION FOR DIFFERENT LINKAGE METHODS AGAINST DIFFERENT DISTANCE METRICS

	ward	median	weighted	complete	single	average
euclidean	0.7733	0.9988	0.9988	0.9981	.9992	**0.9992**
sqeuclidean	0.5707	0.9519	0.9519	0.9461	0.9593	**0.9598**
cityblock	0.7733	0.9988	0.9988	0.9981	0.9992	**0.9992**
mahalanobis	0.7733	0.9988	0.9988	0.9981	0.9992	**0.9992**
minkowski	0.7733	0.9988	0.9988	0.9981	0.9992	**0.9992**
hamming	**0.3605**	0.193	0.193	0.1995	0.184	0.1834
chebyshev	0.7733	0.9988	0.9988	0.9981	0.9992	**0.9992**
jaccard	**0.3605**	0.193	0.193	0.1995	0.184	0.1834

We have also compared the final clusters produced by using this average method with the distance metrics that had high CCC values (Euclidean, Cityblock, Mahalanobis, Minkowski and Chebyshev), and found that their effect on the clusters was negligible since the clusters produced were almost always identical. For this evaluation and the running of the algorithm, a cut-off threshold of 0 was used, which is the maximum inter-cluster distance allowed. Having 0 as the threshold enabled us to form clusters with syntactically identical tags in each cluster, making the next step of expert tagging much more efficient.

B. Usability Evaluation

A usability study is conducted with two sets of participants, experts (mechanical engineers) and non-experts, to gauge the effectiveness of our proposed tagging system on the workload of experts. The inclusion of non-experts in the study gives us a better understanding of the learnability of the system and helps us evaluate whether the previously expert-only task can now be delegated to non-experts. The experiment involves tagging a real dataset of raw sensor names using the developed system. All participants are instructed about the aim of the system and our proposed hierarchy with an example. Their main objective is to use the system to extract as much metadata as they can from the presented dataset.

The study consisted of 3 experts and 2 non-expert users, with the dataset consisting of 28 raw sensor points obtained from the BMS of a local high school. Even a relatively small dataset such as this would traditionally take experts a painstakingly long time to complete. Moreover, the use of this dataset helped to avoid overwhelming the participants during the test, which is likely to influence the usability results of the study. Though the participant size is small (only 5), this would allow us to realise 85% of usability issues [16], [17]. All participants performed the task of mapping the data individually in separate sessions.

The required time to complete the tagging is recorded, and after the study, participants are given the standardised psychometric Post-Study System Usability Questionnaire (PSSUQ) to complete. Standardised post-study questionnaires such as the PSSUQ are increasingly popular due to their many advantages such as replicability, objectivity, quantification, etc. [18], [19]. Although many other standardised usability questionnaires exist (such as

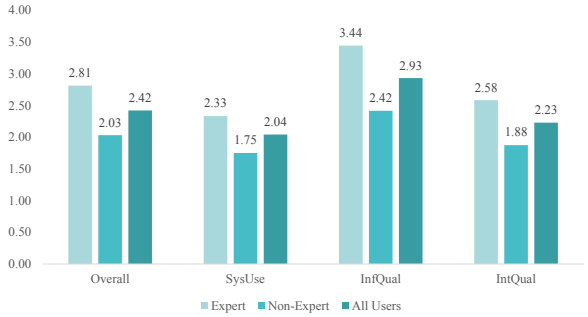

Fig. 8. Summary of the overall score and the sub-scores (SysUse, InfQual, IntQual) for the two types of users, where lower scores indicate a higher degree of perceived usability

SUS, SUMI [18]), the PSSUQ offers 3 additional subscales to rate user's responses; System Quality/Usefulness (SysQual/SysUse), Information Quality (InfoQual), and Interface Quality (IntQual). It comprises of 16 questions, where scores can take any value between 1 and 7 with lower scores displaying a higher degree of user satisfaction. For our study, we used version 3 [18] of the PSSUQ which has been refined over the years to improve its reliability.

The mean scores and sub-scores from the PSSUQ are shown in figure 8, the study as a whole had an overall PSSUQ score of 2.42. Since there have been no similar evaluations completed to compare this set of results against, we will refer to the PSSUQ norms, calculated from 21 studies and 210 participants as a reference point [18].

The AGSDA algorithm, after performing the feature extraction and clustering process, presented to the users with just 2 examples to tag for the whole dataset, and due to this he time taken to complete the whole task of tagging 28 sensors did not take more than 15 minutes for any of the participants. This indicates a substantial improvement in time while using this system in comparison to the traditional methods of manually tagging datasets.

The SysUse and IntQual sub-scores are important since it indicates the actual usefulness of the system developed. And since the algorithm and the ontology developed is independent of the interface, the IntQual can vary depending on the implementation of our proposed methods. The reliabilities for the scores of the PSSUQ are very high, SysUse: 0.9, InfoQual: 0.91, InfQual :083 and Overall: 0.94 [18]. The mean SysUse score of 2.04 from our study is below the norm's (μ = 2.8, 99% CI [2.57, 3.02]) 99% confidence interval and the InfoQual sub-score of 2.93 is below the norm as well (norm: μ = 3.2, 99% CI [2.79, 3.24]).

It can be safe to assume that given the high reliability of SysUse an InfoQual, our study shows that both experts and non-experts had a very high perceived usability regarding the usefulness of the system and quality of the information provided. One question from the questionnaire of significance was, "I believe I could become productive quickly using this system", which had a score of 2.67 from

the experts (norm: $\mu = 2.86$, 99% CI [2.54, 3.17]). This directly fulfils one of the main aims of AGSDA and the proposed ontology; reducing the workload of the experts and increasing their productivity. The non-experts too had good scores and they all managed to complete the presented task which points to the fact that the system can be adopted by users who are not mechanical engineers; which also proves to fulfil the aim of reducing the workload from the experts.

Other scores of the study also indicated high satisfaction in usability when compared with the PSSUQ norms; Overall - 2.81 (norm: $\mu = 2.82$, 99% CI [2.62, 3.02]), IntQual - 2.58 (norm: $\mu = 2.49$, 99% CI [2.28, 2.71])

VII. Conclusion

One of the first tasks in implementing sensor data based solutions is resolving issues related to the semantics of metadata of data points. Our experience indicates that most of the industry participants agree that resolving metadata mapping issues is one of the more time consuming and costly tasks, which often significantly increase the price of the service, negatively impacting on the affordability of such services. It is therefore imperative, from the industry growth perspective, that those issues are addressed and resolved by means of standardised semantics of data points.

In this paper, we propose AGSDA, a novel algorithm that automatically maps heterogeneous BMS sensor data streams into an ontology for further analysis. We also extended a well-known ontology, Project Haystack to fit our proposed hierarchical data mapping and developed a tool that implements AGSDA and the ontology. The tool and the algorithm have been co-designed and developed jointly with our industry partner and is currently integrated into an industry data analytical system where it has been used and tested by industry professionals. Our contributions have undergone both a technical and usability evaluation to deem its efficacy and its ability to reduce the workload on experts analysing BMS data.

Acknowledgement

We would like to acknowledge A/Prof. Weidong Huang from the University of Technology Sydney for guiding us in the usability evaluation for the work.

References

[1] D. Kriksciuniene, T. Pitner, A. Kucera, and V. Sakalauskas, "Data analysis in the intelligent building environment." *IJCSA*, vol. 11, no. 1, pp. 1–17, 2014.

[2] A. Bhattacharya, D. E. Culler, J. Ortiz, D. Hong, and K. Whitehouse, "Enabling portable building applications through automated metadata transformation," *EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2014-159*, 2014.

[3] M. Yalcinkaya and V. Singh, "Building information modeling (bim) for facilities management–literature review and future needs," in *IFIP International Conference on Product Lifecycle Management*. Springer, 2014, pp. 1–10.

[4] J. Fürst, K. Chen, R. H. Katz, and P. Bonnet, "Crowd-sourced bms point matching and metadata maintenance with babel," in *2016 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops)*, 2016, pp. 1–6.

[5] J. Gao, J. Ploennigs, and M. Berges, "A data-driven metadata inference framework for building automation systems," in *Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments*. ACM, 2015, pp. 23–32.

[6] T. Weng, A. Nwokafor, and Y. Agarwal, "Buildingdepot 2.0: An integrated management system for building analysis and control," in *Proceedings of the 5th ACM Workshop on Embedded Systems For Energy-Efficient Buildings*. ACM, 2013, pp. 1–8.

[7] "Project haystack," https://project-haystack.org, accessed: 2019-11-27.

[8] F. Montori, K. Liao, P. Jayaraman, L. Bononi, T. Sellis, and D. Georgakopoulos, in *Web Information Systems Engineering – WISE 2018 - 19th International Conference, 2018, Proceedings*, ser. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), H. Wang, R. Zhou, H.-Y. Paik, H. Hacid, and W. Cellary, Eds. Germany: Springer-Verlag London Ltd., 2018, pp. 209–224.

[9] A. A. Bhattacharya, D. Hong, D. Culler, J. Ortiz, K. Whitehouse, and E. Wu, "Automated metadata construction to support portable building applications," in *Proceedings of the 2Nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments*, ser. BuildSys '15. New York, NY, USA: ACM, 2015, pp. 3–12. [Online]. Available: http://doi.acm.org/10.1145/2821650.2821667

[10] A. Schumann, J. Ploennigs, and B. Gorman, "Towards automating the deployment of energy saving approaches in buildings," 11 2014.

[11] B. Balaji, C. Verma, B. Narayanaswamy, and Y. Agarwal, "Zodiac: Organizing large deployment of sensors to create reusable applications for buildings," in *Proceedings of the 2Nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments*, ser. BuildSys '15. New York, NY, USA: ACM, 2015, pp. 13–22. [Online]. Available: http://doi.acm.org/10.1145/2821650.2821674

[12] "Industry foundation classes (ifc)," https://www.buildingsmart.org/standards/bsi-standards/industry-foundation-classes/, accessed: 2019-11-30.

[13] "Semantic sensor network ontology," https://www.w3.org/TR/vocab-ssn/, accessed: 2019-11-30.

[14] S. Gulwani, "Automating string processing in spreadsheets using input-output examples," in *Proceedings of the 38th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, ser. POPL '11. New York, NY, USA: ACM, 2011, pp. 317–330. [Online]. Available: http://doi.acm.org/10.1145/1926385.1926423

[15] I. Gronau and S. Moran, "Optimal implementations of upgma and other common clustering algorithms," *Information Processing Letters*, vol. 104, no. 6, pp. 205–210, 2007.

[16] W. L. i. R.-B. U. Experience. Why you only need to test with 5 users. [Online]. Available: https://www.nngroup.com/articles/why-you-only-need-to-test-with-5-users/

[17] J. Nielsen and T. K. Landauer, "A mathematical model of the finding of usability problems," in *Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems*, ser. CHI '93. Association for Computing Machinery, pp. 206–213. [Online]. Available: https://doi.org/10.1145/169059.169166

[18] J. Sauro and J. R. Lewis, *Quantifying the User Experience: Practical Statistics for User Research*. Morgan Kaufmann, google-Books-ID: USPfCQAAQBAJ.

[19] J. C. Nunnally, *Psychometric theory*. McGraw-Hill, google-Books-ID: WE59AAAAMAAJ.