Simplifying Data Preparation for Analysis using an Ontology for Machine Data

Dipali Tole and Nikhil Joshi
Asia Technology Innovation Center, John Deere India Pvt. Ltd., Pune, India

Keywords: Ontology, Reasoner, Semantic Web, SQL Query, Database, Automotive Industry.

Abstract: Vehicle manufacturers gather large amounts of data through on-board sensors and other systems, for applications, such as real-time diagnostics, prognostics, design improvements, etc. However, a lot of time is spent in preparing the data for specific analyses. Moreover, this data preparation requires people having expert knowledge about various data schemas and structures used, as well as the specific domain or vehicle systems that the data pertains to. This paper proposes an approach using a formal Ontology to capture knowledge about the domain, and a reasoner to query and prepare data. Using a demonstrative example, the paper presents a comparison of the current approach to preparing data using experts with the proposed approach. The preliminary findings from the study suggest that the proposed approach is promising, and provides unique advantages specifically when faced with distributed, polymorphic data structures, that may change over time.

1 INTRODUCTION

Nowadays, with the proliferation of sensors and electronics in our vehicles and the advances in communication and data storage, vehicle manufacturers have access to large amounts of data from various machines in operation. Analysis of this data can provide valuable insights, such as pre-emptive indicators of failures, load cycles, system performance and security. Any analysis of such data can be broadly considered to constitute three steps:

1. Data Preparation

This includes cleaning the data, removing noise and incorrect data, filtering relevant data, combining data from different sources, resolving conflicts and redundancies, etc. It also involves transforming/ normalizing/ reorganizing data to a form that is required for analysis.

2. Data analysis/modeling

This step refers to the core statistical or analytical tasks, such as determining correlations and dependencies, finding trends and patterns, building mathematical models & visualisations, etc.

3. Interpretation & utilisation

This involves drawing useful insights to support decision making and other intents of the study.

The data required for analysis is usually stored across various databases having inconsistent naming conventions, varying data structures and schemas, and varying conventions for capturing data (such as units, frequency, etc.). Moreover, the data schemas and conventions may also change over time, geography, or with different types of machines. Therefore, there is need for individuals who have intimate knowledge about the domain, implicit assumptions pertaining to data captured, as well as expert understanding of the schemas and database configurations to prepare the data. Such expertise requires years of experience, and even such experienced experts may find it difficult to keep up with the changing data management technologies. A 2016 survey (CrowdFlower, 2016) of data scientists found that up to 79% of their time was spent in data preparation. Moreover, the same survey indicated that 78% of data scientists viewed data preparation as the least enjoyable part of their job.

This paper explores the use of a formal ontology to capture the semantics related to the data, and thereby reduce the efforts and expertise required for the data preparation step. The remainder of the paper is organized as follows: Section 2 introduces a demonstrative example and highlights the issues faced in data preparation without the aid of explicit semantics. Section 3 provides a brief background about ontologies and their applications. It then

Tole, D. and Joshi, N.
Simplifying Data Preparation for Analysis using an Ontology for Machine Data.
DOI: 10.5220/0006892901670174
In *Proceedings of the 10th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2018) - Volume 2: KEOD*, pages 167-174
ISBN: 978-989-758-330-8

describes the creation of a formal ontology, rules, and data instances, for the demonstrative example, and the use of logic based reasoning for data preparation. Section 4 further proposes core common concepts in the form of an ontology for machine information that can be used in the automotive industry for various potential use cases. Section 5 concludes the article with a discussion of the benefits and limitations of the proposed approach, key findings and insights from the study, and directions for future work.

2 DEMONSTRATIVE EXAMPLE

As a demonstrative example, let us consider that we have sensor diagnostic data gathered through telematics for a fleet of vehicles of an OEM. The data would comprise of measurements or diagnostics events from several machines identified by a unique identifier or machine numbers, and generally recorded chronologically. **Figure 1** shows an example of such data in the form of a database table. This database has four fields for every entry, viz. an *EventID* - which provides a unique reference for each data entry, a *DTC* – which provides the diagnostic code that is reported by the machine for the event, a *MachinePin* – which is the unique identifier of the machine, and a *DTCTime* – which records the date and time of occurrence.

EventID	DTC	MachinePin	DTC Time(Date)
1	537689.7	BS208DBJW7584	2/15/17 0:00
2	1048.4	ZW569BDLT9815	2/16/17 0:00
3	1048.4	WH097KITN0193	2/16/17 0:00
4	1048.16	OR358GSDF3802	2/16/17 0:00
5	537689.7	BS208DBJW7584	2/16/17 0:00
6	645.5	WG82THAN9281	2/17/17 0:00
7	4087.15	RW702ENGS8536	2/17/17 0:00
8	1048.16	OR358GSDF3802	2/17/17 0:00
9	537689.7	BS208DBJW7584	2/17/17 0:00
10	1048.3	WG82THAN9281	2/17/17 0:00
11	158.9	EI691FJKH7856	2/18/17 0:00
12	167.9	WH097KITN0193	2/18/17 0:00
13	4087.15	RW702ENGS8536	2/18/17 0:00
14	1999.17	WH097KITN0193	2/18/17 0:00
15	11506.3	CD783JTMO2356	2/19/17 0:00
16	645.5	WG82THAN9281	2/19/17 0:00
17	158.9	EI691FJKH7856	2/19/17 0:00
18	1048.3	WG82THAN9281	2/19/17 0:00
19	1999.17	WH097KITN0193	2/19/17 0:00
20	200.1	NK205FNKH5892	2/20/17 0:00
21	1048.16	DN752HGKM0245	2/20/17 0:00
22	167.9	ZW569BDLT9815	2/20/17 0:00
23	645.5	WG82THAN9281	2/20/17 0:00
24	158.9	EI691FJKH7856	2/20/17 0:00
25	4087.15	RW702ENGS8536	2/20/17 0:00
26	1048.16	OR358GSDF3802	2/20/17 0:00
27	1048.3	WG82THAN9281	2/20/17 0:00
28	200.1	NK205FNKH5892	2/21/17 0:00
29	11506.18	CD783JTMO2356	2/21/17 0:00
30	167.9	ZW569BDLT9815	2/21/17 0:00
31	1048.16	DN752HGKM0245	2/22/17 0:00
32	200.1	NK205FNKH5892	2/22/17 0:00
33	645.6	CD783JTMO2356	2/23/17 0:00
34	1048.16	DN752HGKM0245	2/24/17 0:00

Figure 1: Machine Data Table.

Let us now consider that we wished to analyse this data to find trends in a specific class of diagnostics events, say "hydraulic oil temperature" issues, for a particular type of machine, say "Tractors", over a specific period, say before "20th Feb 2017". In order to analyse the trends, we would need to filter the data to remove information that is not relevant to the study. Clearly, the information available in **Figure 1** is not sufficient to prepare the data for this analysis. We would need the classification of diagnostic codes to identify the *DTC*s that belong to the class of "hydraulic oil temperature" issues, and a catalogue that identifies the type of machine given its unique identifier.

DTC	DTC_Category	DTC_Description
1048.3	Hydraulic Oil Temperature	Hydraulic oil temp sensor input is above operating range(OORH)
1048.4	Hydraulic Oil Temperature	Hydraulic oil temp sensor input is below operating range(OORL)
1048.16	Hydraulic Oil Temperature	Hydraulic oil temperature is above operating range
11506.3	Hydraulic Oil System	Hydraulic oil filter bypass pressure sensor is OORH
11506.4	Hydraulic Oil System	Hydraulic oil filter bypass pressure sensor is OORL
11506.18	Hydraulic Oil System	Hydraulic oil level switch is open
11506.7	Hydraulic Oil System	Hydraulic filter switch is open
200.1	Transmission	Transmission oil temperature greater than operating range
645.5	Transmission	Brake solenoid current less
645.6	Transmission	Brake solenoid current high
645.7	Transmission	Brake Solenoid current is out of command range
537689.7	Transmission	Park brake pressure below normal operating range
158.9	Power System	Battery Power Low
167.9	Power System	Sensor power System Issue
1999.17		Thrown when 1048.4 and 167.9 ocuured together
4087.15		Thrown when 11506.18 and 158.9 occurred together

Figure 2: Classification of Diagnostic Codes.

Figure 2 shows the database table that records the classification of diagnostic codes. It has 3 fields, viz. the *DTC*, a *DTC_Category* – which records the class the *DTC* belongs to and, a *DTC_Description* – which provides a textual description of the issue the *DTC* refers to. Upon close inspection of the last two rows of the database table, it can be observed that the *DTC*s 1999.17 and 4087.15 have not been assigned a *DTC_Category*. This is because they are indicative of a combination of issues as described in the *DTC_Description*. Thus, an event with *DTC*

1999.17 is also indicative of a "hydraulic oil temperature" issue, since it implies occurrence of issue described by *DTC* 1048.4 too. However, this information is not explicit in the data, but implicit in the textual description.

Mach_ID	NativePin	Category	SBU
600	ZW569BDLT9815	MOTORGRADERS	CF
15172	CD783JTMO2356	HARVESTING - COMBINE	HX
81	WH097KITN0193	TRACTORS	TR
231131	RW702ENGS8536	TRACTORS	TR
18239	WG82THAN9281	TRACTORS	TR
568	BS208DBJW7584	HARVESTING - COMBINE	HX
78278	EI691FJKH7856	MOTORGRADERS	CF
20390	NK205FNKH5892	HARVESTING - COMBINE	HX
6804	IW247HDVJ2578	HARVESTING - COMBINE	HX
760034	DN752HGKM0245	MOTORGRADERS	CF
9209	TU8562GHNJ0390	TRACTORS	TR
92	VM538RHNK2403	MOTORGRADERS	CF
91287	UR0249HJKL5025	MOTORGRADERS	CF
546276	BM721TIJF6289	HARVESTING - COMBINE	HX
139012	OR358GSDF3802	TRACTORS	TR

Figure 3: Catalogue of machines in the fleet.

The type of machine can be determined using a separate database table as shown in **Figure 3.** This table has three fields, viz. *MachID* – which serves as a unique identifier of a machine, a *NativePin* –which is a public identifier of the machine and would also be unique to the machine, and *Category* – which records the type of machine. Note that this table uses a different unique identifier for machines to organize its information. Also note that the identifier stored in the field *NativePin* in this table, is the same identifier recorded in the field *MachinePin* in the Machine Data table (**Figure 1**).

Filtering data relevant for the desired analysis, using information in these three database tables requires a series of operations, involving joining of tables based on corresponding key fields, and filtering using desired criteria. This can be achieved using a complex query or series of queries. **Figure 4** shows such a concatenated query written in SQL (Structured Query Language), and the resulting filtered relevant data. The query essentially follows three steps:

i. Filter the database table for the catalogue of machines for entries having the Category "TRACTORS".
ii. Join this filtered table with the Machine Data table using correspondence of NativePin and MachinePin fields, to filter down for events where the correspondence is found (indicating that the events occurred on machines that were "TRACTORS"). Further, filter this table for events that occurred before 20th February 2017 using the DTCTime field.
iii. Finally join this filtered table with the table for Diagnostic codes, using correspondence of the DTC field, and filter for entries having the DTC_Category "Hydraulic Oil Temperature".

As can be evidenced, preparing this data requires expert knowledge of the domain to know equivalence between fields, as well as knowledge of the data structures to know how the information is stored and how it can be manipulated to get the desired information. Moreover, even with such expert knowledge some implicit information may be missed. For example, events with *EventID* 1999.17 are not captured in the prepared data although they indicate a "hydraulic oil temperature" issue. Special queries need to be created to look for such implicit information.

```
MachineDTC=# SELECT eventdata.eventid,machinedata.machine_id,nativepin,cate
gory,eventdata.dtc,eventdata.dtctime FROM machinedata INNER JOIN eventdata
ON eventdata.machinepin=machinedata.nativepin INNER JOIN dtcdata ON eventda
ta.dtc=dtcdata.dtc WHERE machinedata.category='TRACTORS'AND eventdata.dtcti
me <='2017-02-20' AND dtcdata.dtc_category= 'Hydraulic Oil Temperature'ORDE
R BY machinedata.nativepin;
 eventid | machine_id |   nativepin   | category |   dtc   |  dtctime
---------+------------+---------------+----------+---------+------------
      26 |     139012 | OR358GSDF3802 | TRACTORS | 1048.16 | 2017-02-20
       4 |     139012 | OR358GSDF3802 | TRACTORS | 1048.16 | 2017-02-16
       8 |     139012 | OR358GSDF3802 | TRACTORS | 1048.16 | 2017-02-17
      18 |      18239 | WG82THAN9281  | TRACTORS |  1048.3 | 2017-02-19
      27 |      18239 | WG82THAN9281  | TRACTORS |  1048.3 | 2017-02-20
      10 |      18239 | WG82THAN9281  | TRACTORS |  1048.3 | 2017-02-17
       3 |         81 | WH097KITN0193 | TRACTORS |  1048.4 | 2017-02-16
(7 rows)
```

Figure 4: Concatenated SQL query and resulting output.

3 DATA PREPARATION USING ONTOLOGIES

To overcome the issues identified in Section 2, we propose an approach involving the creation of a domain ontology, and performing logic based reasoning, to assist in the data preparation.

In computer science, an Ontology is a formal, explicit specification of the concepts, relationships, and other distinctions that are relevant for modelling a domain (Gruber, 2009). It provides a common vocabulary, usually machine-interpretable, to share a common understanding of the structure of information among people and software agents and helps make domain assumptions explicit (Noy & McGuinness, n.d.). It thereby allows software agents, often called reasoners, to identify implicit information in the data based on first-order logic. Such reasoners have been used to enable interoperability between software tools, determine inconsistencies and errors in data, automate data classification, etc. (Ameri, et al., 2012) (Yang, et al., 2013).

We shall explain our proposed approach using the demonstrative example introduced in Section 2. For this example, we use an Ontology Editor, Protégé (Musen, 2015), provided by the Stanford Center for Biomedical Research, Stanford University. Protégé supports OWL-DL (Web Ontology Language – Description Logic) as the language for defining the Ontology. It enables reasoning using Description Logic, which is a subset of first-order logic (Horridge, 2011) (Wood, 2013). We also use the Pellet reasoner (Clark, 2015) plugin for Protégé for drawing inferences.

To define the Ontology, we first identify the important concepts in the domain. In this example, the key concepts are that of a **Machine**, a diagnostic code or **DTC**, and a diagnostic **Event**, which are defined as classes in the Ontology. Every machine has a pin number and type or category. To capture that, we create two new classes **nativepin** and **MachineCategory**, and two new relationships or object properties, namely **hasNativePin**, which has **Machine** as the domain and **nativepin** as the range, and **hasCategory**, which has **Machine** as the domain and **MachineCategory** as the range. Similarly, for a **DTC** we create a new class, **DTCCategory**, an object property, **hasFaultCategory**, and a data property **hasdescription**. Finally, for the **Event** class we create a class, **machinepin**, an object property, **hasMachinePin**, an object property **hasFaultCode**, and a data property, **date**. The domains and ranges for these properties are shown in **Table 1** below:

Table 1: Table of relationships.

Property	Domain	Range
hasCategory	Machine	MachineCategory
hasFaultCategory	DTC	DTCCategory
hasFaultCode	Event	DTC
hasMachinepin	Event	machinepin
hasNativePin	Machine	nativepin
date	Event	<dateTime>
dtcdescription	DTC	<string>

As we analyses the domain, we realise that the classes **nativepin** and **machinepin** describe the same concept, and hence we specify that these classes are equivalent. Likewise, we explicitly specify all the other classes to be disjoint from each other. Since, we know that diagnostic events occur on specific machines, we would like to have a relationship that indicates the **Machine** that a particular **Event** occurred on. Therefore, we create a new object property, **belongsTo**, with **Event** as the domain and **Machine** as the range. However, we also realise that this information would implicitly be present in the data through the **hasMachinePin** and **hasNativePin** properties due to the equivalence of **machinepin** and **nativepin**. Hence, we specify the property **belongsTo** as SuperProperty Of Chain "**hasMachinePin** o *inverse*(**hasNativePin**)".

The resultant description of the domain can be visualized as shown in **Figure 5**.

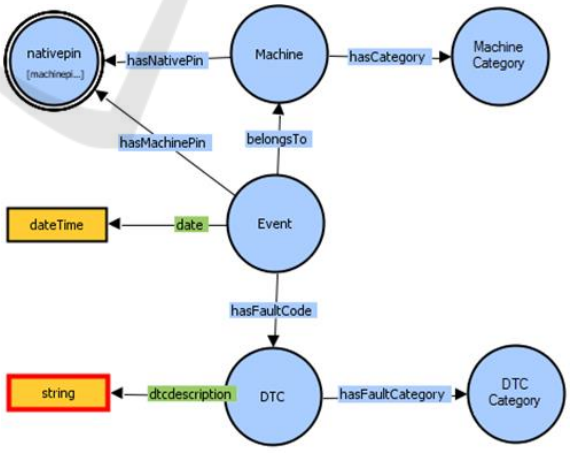

Figure 5: Definition of the machine data ontology.

The information specified so far is merely recording knowledge about the domain. This information is often referred to as the T-Box or Terminology Box. It does not have any information about specific instances of machines or specific diagnostic events.

That data, which is available in database tables as shown in **Figure 1**, **Figure 2**, and **Figure 3** also needs to be specified. This information is often referred to as the A-Box or Assertion Box. For our example, we have prepared the database tables in Excel, and use the Cellfie plugin (Hardi, 2016) to create axioms specifying the A-Box in the Ontology.

It should be noted that data in any form (e.g. SQL/JSON database) can be imported into the ontology with the use of an appropriate mechanism. The resultant knowledge base would now have instances for each of the concepts or classes along with relationships that are explicitly present in the database tables. **Figure 6** is a screenshot of the Ontology in Protégé showing an instance *14* of type **Event** that **hasMachinePin** *WH097KITN0193*, **hasFaultCode** *1999.17*, and has **date** *2017-02-18*.

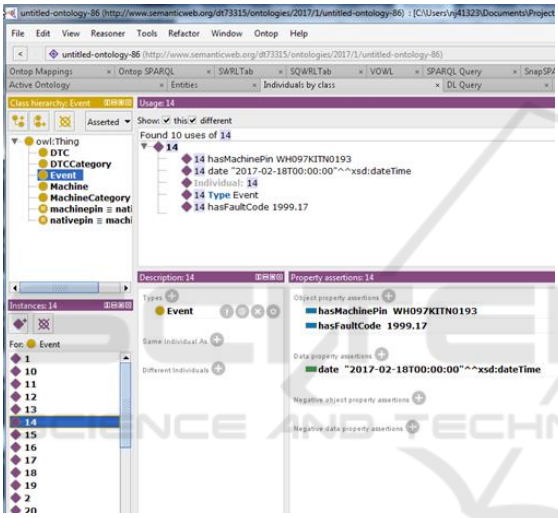

Figure 6: Screenshot of Protégé showing an instance of class Event.

In order to capture the implicit information about certain diagnostic codes that indicate a combination of issues, we can use rules that indicate the implied axioms. For example, we can create a rule that specifies that if there exists a relationship **hasFaultCode** between an **Event** *E* and **DTC** *1999.17*, it implies that there also exist **hasFaultCode** relationships between the **Event** *E* and **DTC**s *167.9* and *1048.4*. **Figure 7** shows the rules captured in the Ontology.

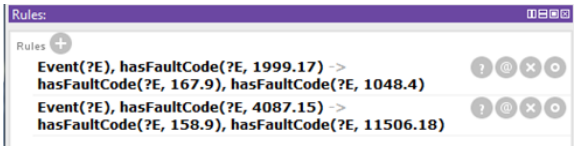

Figure 7: Rules capturing implicit information.

Once we have the information captured in the Ontology we can start the reasoner software to infer implicit information in the data. For example, as shown in **Figure 8**, the reasoner has inferred three more axioms or relationships for **Event** *14*, viz. **hasFaultCode** *167.9 (***DTC**), **hasFaultCode** *1048.4* (**DTC**), and **belongsTo** *81* (**Machine**) as highlighted in orange box.

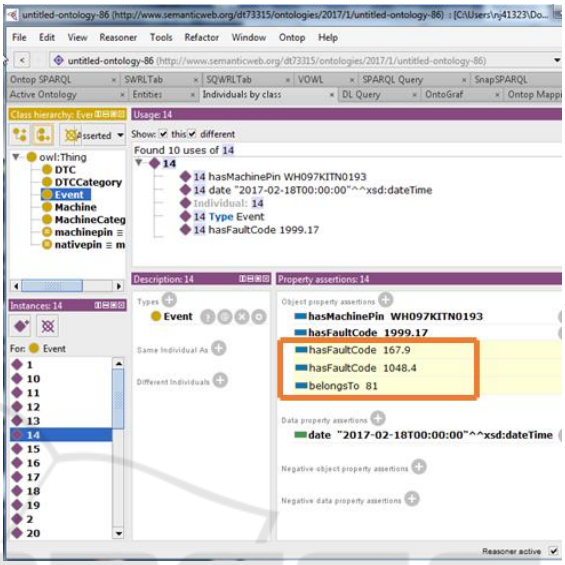

Figure 8: Screenshot showing inferred information.

As can be seen, with the use of the Ontology and reasoning, we would have rich explicit information which can aid in data preparation for various applications. In our example, we are looking for all diagnostic events of a specific type, viz. "hydraulic oil temperature" issues that have occurred on a specific type of machine, viz. "Tractors", over a specific period, viz. before "20th Feb 2017". To prepare this data we use the Snap-SPARQL (Musen & Horridge, 2015) query language. The definition of the query is shown in **Figure 9**. As can be seen from the results of the query in **Figure 10** diagnostic events that included "hydraulic oil temperature" issues implicitly but not explicitly in the data, highlighted in a green box, are also identified. The resulting data can be used for the desired analysis, and no additional data preparation tasks are required.

As can be seen in **Figure 9**, the query can be built at conceptual level, including concepts as defined in the T-Box. Consequently, knowledge of the domain and the needs of the application is all that is required to carry out the data preparation task. Knowledge of the different data structures used for different types of information is not required. Knowledge of database manipulation techniques, joining, filtering and manipulating database tables is

```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX eg:<http://www.semanticweb.org/dt73315/ontologies/2017/1/untitled-ontology-86#>

SELECT ?eventid ?machineid ?Pin ?machinecategory ?dtccode ?date
WHERE{ ?eventid eg:hasFaultCode ?dtccode;
                eg:date ?date;
                eg:belongsTo ?machineid.
       ?machineid eg:hasCategory ?machinecategory;
                  eg:hasNativePin ?Pin.
       ?dtccode eg:hasFaultCategory eg:HydraulicOilTemperature.
       FILTER ( ?date <= ("2017-02-20T00:00:00" )
       && ?machinecategory = eg:TRACTORS) }
ORDER BY ?machineid
```

Figure 9: SnapSPARQL query to prepare data.

?eventid	?machineid	?Pin	?machinecategory	?dtccode	?date
eg:26	eg:139012	eg:OR358GSDF3802	eg:TRACTORS	eg:1048.16	2017-02-20T00:00:00^^xsd:dateTime
eg:4	eg:139012	eg:OR358GSDF3802	eg:TRACTORS	eg:1048.16	2017-02-16T00:00:00^^xsd:dateTime
eg:8	eg:139012	eg:OR358GSDF3802	eg:TRACTORS	eg:1048.16	2017-02-17T00:00:00^^xsd:dateTime
eg:10	eg:18239	eg:WG82THAN9281	eg:TRACTORS	eg:1048.3	2017-02-17T00:00:00^^xsd:dateTime
eg:27	eg:18239	eg:WG82THAN9281	eg:TRACTORS	eg:1048.3	2017-02-20T00:00:00^^xsd:dateTime
eg:18	eg:18239	eg:WG82THAN9281	eg:TRACTORS	eg:1048.3	2017-02-19T00:00:00^^xsd:dateTime
eg:14	eg:81	eg:WH097KITN0193	eg:TRACTORS	eg:1048.4	2017-02-18T00:00:00^^xsd:dateTime
eg:19	eg:81	eg:WH097KITN0193	eg:TRACTORS	eg:1048.4	2017-02-19T00:00:00^^xsd:dateTime
eg:3	eg:81	eg:WH097KITN0193	eg:TRACTORS	eg:1048.4	2017-02-16T00:00:00^^xsd:dateTime

9 results

Reasoner active ☑ Show Inference

Figure 10: SnapSPARQL query results-prepared data.

also not required, although the user needs to be able to interact with the knowledge base using appropriate querying language and interface. Moreover, since implicit information can be explicitly specified in the knowledge base, it is not necessary to make separate considerations during the data preparation task.

4 COMMON ONTOLOGY FOR MACHINE INFORMATION

While the demonstrative example introduced in Section 3 introduces the approach and the benefits of using a domain ontology, the ontology developed for the example is inadequate for capturing key concepts about machine data. In this section, we identify some core concepts and their relationships, towards building a more complete ontology of Machine information for the automotive industry.

The developed ontology is shown in **Figure 11**. As shown in the figure, the ontology captures five aspects of machine information, viz. the structural and product hierarchy as envisaged by the enterprise for their family of products, sensor and measurement data for a machine, service and maintenance records, diagnostic event data, and warranty event data.

Such an ontology would enable users to capture key concepts of machine information. Depending upon the information captured and configured in various databases, it may be necessary to include additional concepts and relationships. In some cases, the concepts and relations mentioned in the ontology may not be explicit in the data and would need to be inferred using reasoners. It is typical for the enterprise to manage these five aspects of machine information in separate databases. Use of such a complete ontology, with appropriate mappings for the databases, will enable complex analysis of machine data. For example, one could study trends in sensor measurements or occurrence of diagnostic trouble codes in a period immediately preceding specific class of warranty event, or identify correlation between diagnostic events and sensor measurements.

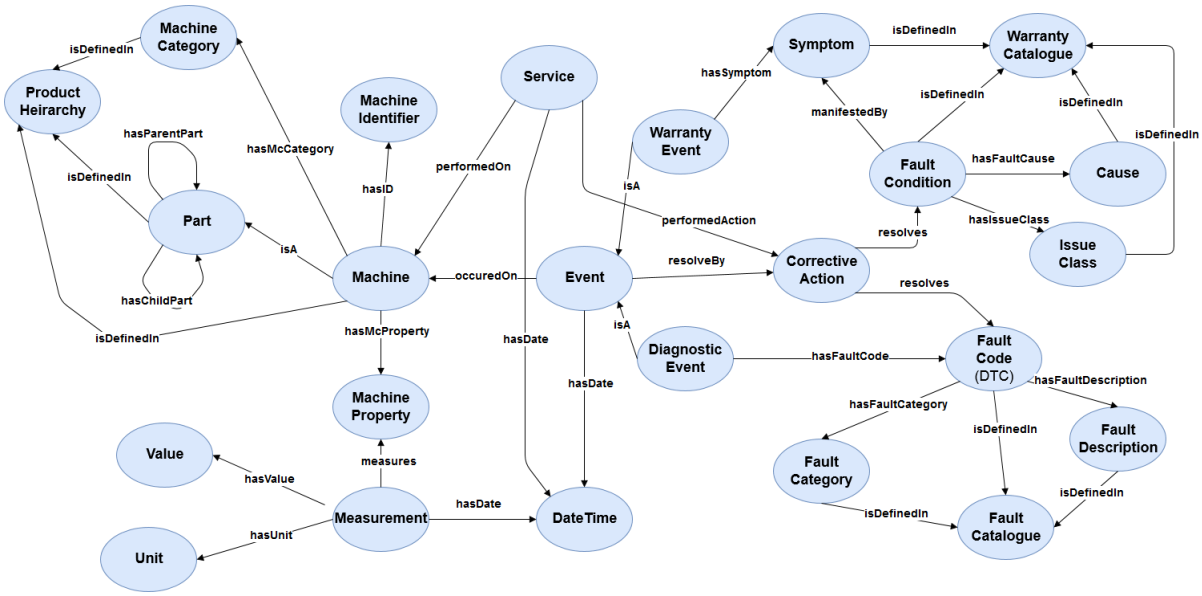

Figure 11: Preliminary core ontology of Machine Information for Automotive industry.

5 CONCLUSION

In this paper, we have used a demonstrative a domain ontology for machine information, along with mapped data and reasoning to simplify knowledge tasks. We have also presented a set of core concepts that would be useful for developing a more detailed ontology of machine information across the automotive industry.

The proposed approach is expected to be advantageous when performing analysis across multiple and polymorphic data sources that capture information of sensors, or reporting different types of diagnostic events, etc. The proposed approach separates the activities of maintaining the equivalence between concepts, mapping of data fields to concepts, and preparing data for specific analytics activities. Thus, expert knowledge about the evolution of the data structures, may not be needed on the part of the individual performing data preparation tasks.

The proposed method, therefore, shows benefits in terms of reduction in the effort and expertise required to perform the data preparation tasks. However, the approach is computationally more expensive, due to the additional logic based computation needed from the reasoner. Also, the domain information and the mapping between the database tables and the concepts defined in the Ontology, needs to be maintained, while the domain as well as the databases evolve over time.

Finally, the proposed approach also has the potential to enable merging concepts from multiple domains, e.g. diagnostic data and warranty data, identifying overlapping concepts and relationships, and reasoning over combined datasets to draw insights on the complex dependencies and behaviours that are implicit in the data. Present day vehicles are complex assemblies involving sub-systems developed by different vendors. Therefore, the authors argue for development of standard ontologies for machine information. This would be highly advantageous in drawing useful knowledge and insights from the machine information being collected.

REFERENCES

Ameri, F., McArthur, C. & Urbanovsky, C., 2012. *A Systematic Approach to Developing Ontologies for Manufacturing Service Modeling.* Graz, Austria, s.n.

Clark, K., 2015. *Pellet Reasoner.* [Online] Available at: https://github.com/stardog-union/pellet [Accessed March 2018].

CrowdFlower, 2016. *2016 Data Science Report,* s.l.: CrowdFlower.

Gruber, T., 2009. Ontology. In: L. Liu & M. T. Özsu, eds. *Encyclopedia of Database Systems.* US: Springer US, pp. 1960-1976.

Hardi, J., 2016. *Cellfie plugin.* [Online] Available at: https://github.com/protegeproject/cellfie-plugin/wiki [Accessed 6 Mar 2018].

Horridge, M., 2011. A Practical Guide to Building OWL Ontologies Using Protégé 4 and CO-ODE Tools Edition 1.3.

Musen, M. A., 2015. The Protégé project: A look back and a look forward. *AI Matters*, June, 1(4), pp. 4-12.

Musen, M. & Horridge, M., 2015. *Snap-SPARQL: A Java Framework for working with SPARQL and OWL*. Bethlehem, PA, USA, s.n.

Noy, N. F. & McGuinness, D. L., n.d. *Ontology Development 101: A Guide to Creating Your First Ontology*, s.l.: Stanford University.

Wood, M., 2013. *Semantic Web, OWL & Protégé*. [Online] Available at: https://docslide.us/documents/semantic-web-owl-protege.html [Accessed 3 March 2018].

Yang, K., Kim, W., Yang, J. D. & Kim, Y. K., 2013. *Ontology Matching for Recommendation of HS Code.* Hong Kong, s.n.