Check for updates

Proceedings of the Institution of Mechanical Engineers, Part B:
Journal of Engineering Manufacture, Available online 09 December 2020

*Special Issue Article*

Institution of MECHANICAL ENGINEERS
Journal of ENGINEERING MANUFACTURE

*Proc IMechE Part B:*
*J Engineering Manufacture*
1–15
© IMechE 2020
Article reuse guidelines:
sagepub.com/journals-permissions
DOI: 10.1177/09544054209781 17
journals.sagepub.com/home/pib
SAGE

# Data management for developing digital twin ontology model

**Sumit Singh[1], Essam Shehab[1,2], Nigel Higgins[3], Kevin Fowler[3], Dylan Reynolds[3], John A Erkoyuncu[1] and Peter Gadd[3]**

**Abstract**

Digital Twin (DT) is the imitation of the real world product, process or system. Digital Twin is the ideal solution for data-driven optimisations in different phases of the product lifecycle. With the rapid growth in DT research, data management for digital twin is a challenging field for both industries and academia. The challenges for DT data management are analysed in this article are data variety, big data & data mining and DT dynamics. The current research proposes a novel concept of DT ontology model and methodology to address these data management challenges. The DT ontology model captures and models the conceptual knowledge of the DT domain. Using the proposed methodology, such domain knowledge is transformed into a minimum data model structure to map, query and manage databases for DT applications. The proposed research is further validated using a case study based on Condition-Based Monitoring (CBM) DT application. The query formulation around minimum data model structure further shows the effectiveness of the current approach by returning accurate results, along with maintaining semantics and conceptual relationships along DT lifecycle. The method not only provides flexibility to retain knowledge along DT lifecycle but also helps users and developers to design, maintain and query databases effectively for DT applications and systems of different scale and complexities.

**Keywords**

Digital twin, data modelling, ontologies, data management

Date received: 7 April 2020; accepted: 8 November 2020

## Introduction

Digital Twin (DT) is the combination of logically integrated models of a physical asset to give useful insights using data associated with those models. The concept DT has been introduced by Grieves and Vickers[1] at the University of Michigan in 2002 refereeing it as the conceptual ideal for the Product Lifecycle Management (PLM). DT is predicted to play a significant role in improving consistency, seamless development process and the possibility of reuse in subsequent stages along the product lifecycle.[2] DT is built on three main pillars: (a) a physical product in real space, (b) a virtual product in virtual space and (c) the connection of data and information which ties together both the spaces.[3] Today, the lower costs and improved power and capabilities resulted in leaders to combine Information Technology (IT) and Operational Technology (OT) to enable the creation and use of DT.[4] The principle approach of the DT by Boschert and Rosen[2] explains how DT uses digital information from IT systems, such as PLM, PDM (Product Data Management), SCADA (Supervisory Control and Data Acquisition), and makes it available for phase-specific analysis within the lifecycle. DT encapsulates software object/model that mirrors physical asset, and perform analytics based on this digital information. Ideally, this information is based on product-related and historical data along with the enterprise system. A well defined DT can considerably improve decision making in the enterprise at various level of complexity and scale. As they are lined or linked to their physical counterparts, are used to analyse the state of the product or system or process, respond to the changes, improve operations and, add value to the overall enterprise atmosphere.

[1]Cranfield University, Cranfield, Bedford, UK
[2]Nazarbayev University, Nur-Sultan, Kazakhstan
[3]Airbus Operations Ltd., Filton, Bristol, UK

**Corresponding author:**
Sumit Singh, Cranfield University, College Road, Cranfield, Bedford MK43 0AL, UK.
Email: sumit.singh@cranfield.ac.uk

As the concept of DT is novel, there are several challenges exist in its development and implementation. Such challenges have been summarised in Singh et al.[5] for present high-value manufacturing industries under five different themes. The existing research focusses on the data management challenges for DT. High-value manufacturing industries, such as aircraft manufacturing industry, often evolve and perform in the complex data ecosystem. The data management around DT is not explored well in the existing literature. Only a few research works[6–12] focus on the information and data side of DT. Data management is an essential part of any software project so thus becomes essential for DT for different order scale and applications. Therefore, it is important to understand not only the minimum level of a data structure but also knowledge to represent it. Such knowledge will help in understanding data flow, data properties and constraints that DT may exert on the overall system and databases. In this paper, an approach based on ontology and data modelling is proposed to address such data-related issues for DT. The approach is further validated using a case study which reflects the potential for wider applications.

The key contributions of this paper are (a) understanding the information flow and need of efficient data management for a concept like DT, (b) novel method to propose minimum data structure to model data for DT, (c) use of ontologies to define semantics, restrictions and data structure for DT to domain application, (d) accessibility to the user to the most significant data to query databases using directly from DT domain ontology. The contributions are validated through a case study based on Condition-Based Monitoring (CBM) of the asset. The rest of the paper is organised as follows: Section II discussed the data management challenges for digital twin, section III presents the state-of-the-art related works in data modelling, ontology modelling and knowledge graphs in digital twin context. The sections further describe the digital twin ontology model and the methodology, section IV details a standard based CBM application case study, section V is the analysis of results, and final section VI & VII concludes the paper with discussion, and conclusion and future work respectively. The current work is the extension of existing work.[13]

## Data management challenges for Digital Twin

Data management is one of the important branches of developing and maintaining almost all variety of information system. In this scenario, data management challenges are for comprehensive DT solutions becomes obvious. The current trend of transformation from production-oriented to selling services as products is becoming common in the current industrial landscape. This leads to new paradigms of product definition and development. One of the greatest assets in this shift is data. Data-driven solutions are driving innovations and value creation in almost every industry. DT utilised this data to define boundaries of physical and virtual systems to simulate and optimise existing products, process and systems. Although DT has a huge potential of optimising current businesses, the issues related to data is much more complex. To understand the data issues, understanding the data management challenges for DT is the key. Data management is a classic problem of existing systems from product design to asset management and maintenance. The following issues make the data management for DT difficult:

### *Data variety*

The current manufacturing industries generate a massive variety of data across the product lifecycle. Starting with product development, design data in terms of 2D, 3D drawings are very different from the Finite Element Analysis (FEA) and other simulation data. The manufacturing data is structured in completely different formats from design and engineering. The systems like PLM and SAP/ERP can be considered as an organised form of such a wide variety of data. There still lack a bridge that how these systems can be used for a single integrated platform as DT. Such a large variety of data raises the data integration, data cleansing and data fusion issues.[3] The existing DT research shows that most of the DT are application-centric. For example, utilisation of machine data for shop-floor based DT, damage tolerance data to build structural based twins, but lack ways of integrating them into one. This may hinder the development of ideal integrated multiple architectures and frameworks for DT. Some of the scenarios are even hard to capture, for example, shop floor uncertainties can only be managed based on individual user's experience and situational response to uncertainties. This form of knowledge is hard to record or store digitally.

### *Big data and data mining*

The challenges of big data and its mining goes hand in hand. The data collected from various streams in product development and manufacturing, it needs to be stored in databases, accessed and processes to valuable information. In DT context, this data becomes key for the virtualisation of the physical asset. Data mining is the way of finding useful patterns from data sources,[14] therefore it is potentially a key factor for improving the virtual spaces in DT. A large variety of data during product lifecycle results in bigger and complex databases making data mining difficult.[14] Data mining in some industries such as manufacturing is limited to 10% for solving problems by applying data mining techniques. DT models work on continuous improvement of virtual models based on real-time and historical data but data mining techniques are limited to production, fault diagnosis and maintenance phases of the product lifecycle. Data mining for the converging behaviour of physical and virtual spaces in DT is still an open area of research.

The convergence of big data is one of the major problem related to data in the DT context. Data governance is an important part of the big data project.[15] One of the major problem related to data in DT. Big data involves the collection of data sets that are so large and complex that it becomes difficult to process using hands-on database management tools or traditional data processing applications. DT linked to big data acquisition, processes and analytics convergence of increasing data generated results in storage-related issues.[16] The 6 V's definition of big data given by Tyagi and Demirkan[15] gives a clear picture of the data. The concepts of data lakes have been recently welcomed by enterprises to over big data issues as low cost. Using data lakes, enterprises can perform better data management transformation, processing and analytics around a specific application. Even though data lakes provide a new paradigm in data ingestion, transformation, federation and data discovery, it still lacks data governance and technology integration reforms.

### *Digital Twin dynamics*

DT dynamics covers multiple aspects of uncertainty, the exactness of virtualisation and continuous update of data and information between physical and virtual spaces. The close one-to-one mapping between physical and virtual system is incomplete without acknowledging uncertainties involved in both physical observations and digital models.[17] Uncertainties quantification is not only important for giving reliable results but also important for the evolution of DT over time. Several uncertainties parameters are outlined by Kennedy and O'Hagan.[18] The exactness of virtualisation is another important and challenging element for DT. This refers to the degree of exactness the physical asset is imitated in its virtual system. No research claims the degree of exactness can be fully achieved but measuring techniques can be used to measure based on the application.

The continuous update of data and information between physical and virtual systems is fundamental for DT dynamic behaviour. The integration of data from sensors and machines, historical data and computer-based models leads to directional analysis and visualisation of the results. In this process, the continuous chain of incoming data and change in historical data is desirable for keeping the results from DT up-to-date. The important questions are how the continuous update of data leads to data management difficulties. Each DT simulation cycle can lead to changes in existing data repositories and data structures. Maintaining such data repositories and structures brings data management challenges for DT. Connectivity via IoT solutions is another important aspect. Semantic-based data modelling and knowledge graphs are promising methods of simplifying complexity around continuous updating behaviour of a DT.

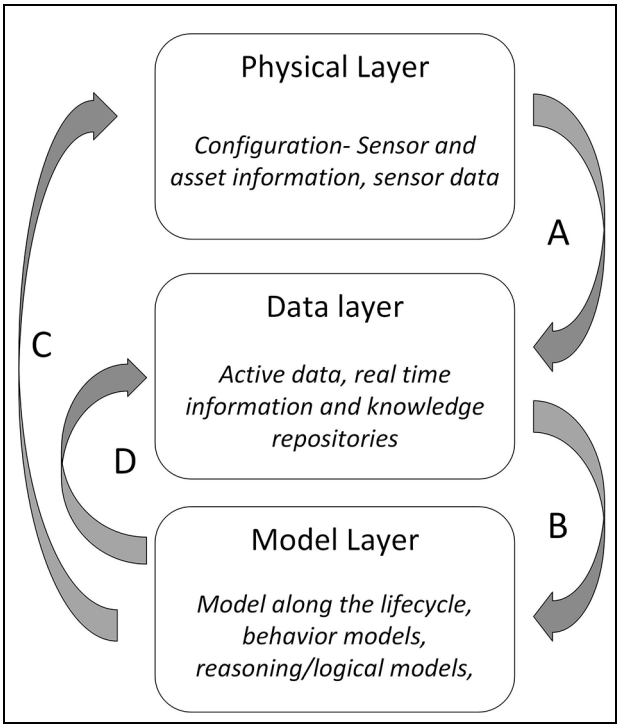


**Figure 1.** DT information flow.

## Digital twin, data modelling and ontology

### *Digital Twin*

The DT market is estimated to grow from USD 3.8 billion in 2019 to USD 35.8 billion by 2025.[19] With the advances in technologies such as the Internet of Things (IoT)[20] and cloud,[21] more companies are willing to adopt DT technology at different levels. As DT is heavily driven by information across the asset/system, understanding the flow and transfer of information at each step becomes one of the key aspects. With an information point of view, we defined the multi-layer information flow across DT. As shown in Figure 1, the information flow establishes among each layer contain a different set of information that can complete the information cycle along with the DT. The information flow steps are defined as follows:

A. Physical layer to data layer: The physical layer denotes the physical entities of the DT such as asset. The physical layer is restricted to configuration information related to asset and raw sensor data. Configuration information is used as a signature throughout DT lifecycle whereas raw sensor data is further filtered and manipulated in the data layer. The configuration information provides traceability[22] and helps in information organisation across DT.
B. Data layer to model layer: The data layer denotes the data fit for analysis and knowledge repositories. The knowledge repositories hold business rules,

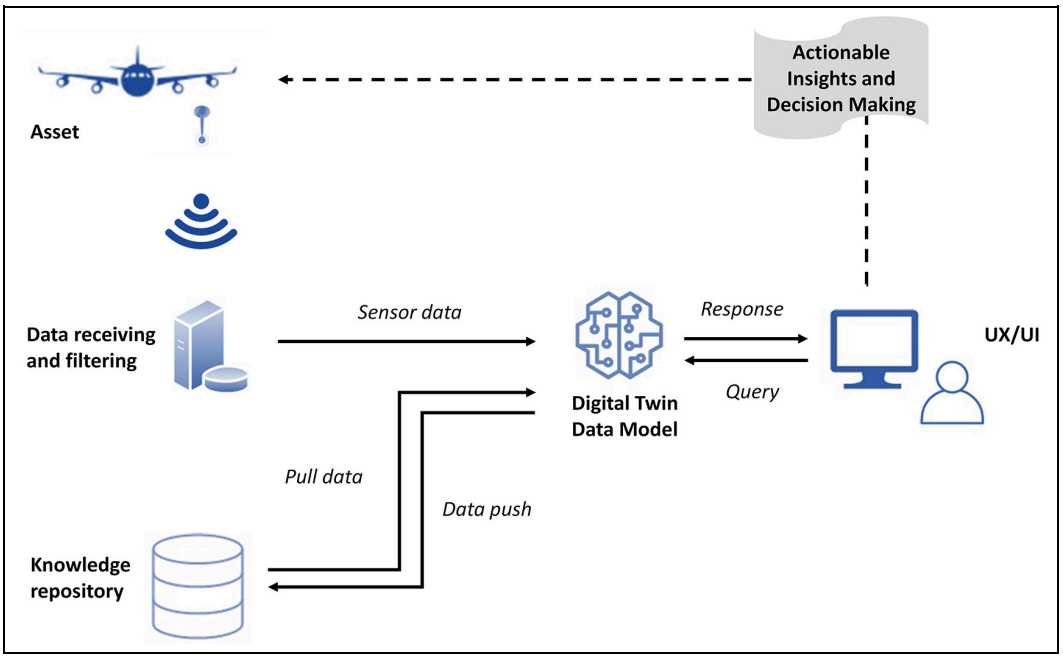


**Figure 2.** DT data query and response mechanism.

logics and historical information about the asset. This manipulated sensor data is not only fit for models to run simulation but also increases understandability and accuracy. The knowledge repository supply historical data and business rules to model layer to provide valuable insights about the asset.

C. Model layer to physical layer: The model layer holds model along the lifecycle, behaviour model and logical models for reasoning. These models use information from the data layer to provide actionable insights by predicting failures and detecting the current state. These insights are implemented in the physical layer as a maintenance operation and service strategy.

D. Model layer to data layer: The information generated in the model layer is stored back to the knowledge repositories in the data layer. Such information contain analysis reports and recommendations that need to be used for the next DT analysis cycle. Such information serves back the model layer as historical data.

Based on such information flow, managing and organising data is crucial for DT along with this flow. A user should be able to run queries to understand the current state of the asset w.r.t. to different variables. Such flexibility can be achieved by a well-structured data model. As shown in Figure 2, the DT data model is an important gateway between the data coming from the physical layer and knowledge repository to produce actionable insights and overall decision making.

## Data and ontology modelling

Data modelling is the first step in the process of database design. Data models describe relevant concepts, data structures from an application and inscribe useful knowledge useful to drive the application behaviour. Although data modelling is widely used in industries for more than three decades but often lacks semantics during the development process. The issues related to heterogeneous databases and interoperability are often hard to manage with data models. To overcome such issues, semantic web technology is known for new ways of managing data and metadata maintaining a higher order of logical and conceptual schemas. Semantic web enables an open-world oriented integration of diverse data sources that uses distributed incantations of closed world data dictionaries, schemas and inference rules. Resource Description Framework (RDF) is widely recognised as one of the technologies for semantic web. RDF has real potential when properties and values in the domain are defined by shared schema or ontology. An ontology is the explicit formal specification of concepts in the domain and relations among them.[23] An ontology defines a common vocabulary for researchers who need to share information in a domain.[24] Ontologies provide building blocks of RDF based conceptual models by providing a formal definition of a set of concepts within a domain and the relationships between those concepts.

*Use of ontologies in data management.* Ontologies can be used as a potential guide to validate the domain models by allowing interaction between data held in different formats. Ontology models contain the concept definition and their relationship to a particular domain. This also includes domain rules such as cardinality, disjointness, etc. that restricts the semantic concepts and the conceptual relationships in a specific conceptualisation of particular application domain. DT requires a

comprehensive approach to query data from information systems. Although ER models are used primarily for database design, they often do not store domain knowledge. Therefore, ER based query formulation approaches[25] can not provide reliable approaches to satisfy the level of comprehensiveness for data queries. Therefore looking at the challenges of effective DT data management. Plenty of research has emphasised on Ontology to database mapping and Database to ontology transformation approaches. A review of such work has been conducted by Munir and Anjum.[25] Although both methods are important while linking ontologies to the database design but have their challenges:

- Most of the ontology to database mapping work on the assumption that both database and ontology pre-exists, and produce a set corresponding mapping between the relational database schema and ontology schema.
- The database-to-ontology transformation approaches are only effective when only a relational database already exists and an ontology is produced by applying certain transformation rules.[26] Most of these practices result in ontologies with a flat structure as the original database.

Considering the issues of continuous update of data and information in DT lifecycle, choosing the right transformation and mapping technique is required. Therefore, the approach of ontology to the data model is adopted in the current research. The ontology-to-conceptual data model is used as the core of the five-step methodology. El-Ghalayini et al.[27] proposed mapping rules that guide from domain ontology to the corresponding schema of the conceptual data model (CDM). Another method to transform domain ontology into a relational database is investigated by Vysniauskas and Nemuraite[28] based on algorithms embedded in OWL2DB.[29] In this approach, OWL document analysed to generate corresponding Data Definition Language scripts. During this analysis and data transformation, the system first transforms ontology classes to data table definitions. Further, the objects, data type properties and constraints into a DDL statement and finally database is filled with class instances. The approach uses a breadth-first search on the hierarchical levels of ontology classes[29] creating a one-to-one mapping between their classes and subclasses. The OWL object properties use again breadth-first search to transform into table relationships. Based on the defined cardinality of class properties, one-to-many or many-to-many relationships between tables are generated.

The proposed DT ontology contains the definition concepts and their relationships for specific DT application including assertions and domain rules, cardinality, disjointness, that restricts semantics of concept and conceptual relationship in specific conceptualisation in a specific domain. Therefore, the proposed DT ontology model contains the conceptual knowledge of DT domain.

Ontologies are semantically richer than database schemas. Database schemas only target to establish relationships between users and domain requirements and describe logical structure of the data.

*Graphs versus relational databases.* In relational databases, to analyse relationships across different table entities, the time expensive 'join' operation is used to combine the relations. This operation is expensive as it requires index lookups and matching to related columns in the tables. This set major drawbacks of graphs. On the other hand, graphs store entities and their relationships as nodes and edges that may be augmented at different attributes retrieving the edge between two entities do not involve expensive 'join' operation.

But graphs have their complexities while using it for real-world applications and using with legacy IT systems:

- The complexity of using graph language- currently the well-known graph query commands are Gremlin, cypher and SPARQL. Each language uses a different approach to querying the database. Casual users may find it difficult to use it on the first hand. Therefore, users require to query the system directly without worrying about learning the new syntax of the unfamiliar query language.[30] Graph databases are language-specific and have their APIs.
- Multi-user support- relational databases in general and relational databases specifically have extensive built-in-multi user support. On the other hand, many graphs based approaches lack support for a multi-user environment. Neo4j uses cypher based query language that forces all user management to be handled at the application level.[31]
- The security aspect is further discussed by Vicknair et al.[31] and Miller.[32] Relational databases such as MySQL contains extensive support for Access Control List (ACL)-based security. Neo4j does not have any built-in-security support.[31] Although Neo4j website does contain some rudimentary design for an access control list (ACL) security mechanism, like multi-user support, ACL management is handled only at the application level.

The graph databases lack standardisation on language for transversal and insertion. This leads to vastly different implementations and framework for data interaction. There is a lack of consistency that requires one to learn all implementations before understanding the appropriate approach suitable to the problem.[32] The decision of choosing graph-based versus relational

databases is driven by system requirements. Since the existing applied research is more focussed on developing a solution to support multi-user environment and standardised language to query data & information to each user, the use of graph-based databases is excluded in this research scope. If the system requires dynamic data modelling that represents highly connected and complex data, the graph-based approach is significant.[32] GQL can be used as the future work of the existing research when the consistency of information retrieval from the integration of multi-domain DTs be explored.

### Related work

Some research works used ontologies in DT domain but has their drawbacks when comes to highly scalable and interoperable application. An ontology-based design framework for co-evolution with complex engineering system by capturing data in terms of variety, velocity and volume.[33] Bao et al.[34] proposed an ontology-based framework to model assembly oriented part DT. The framework demonstrates the main components and dataflow in creating a part DT with information filtering and subsequent management. Mehdi[35] used ontologies for querying data and semantically integrate knowledge base to facilitate intelligent diagnostics of an industrial turbine. Banerjee[36] proposed a way of formalising knowledge as DT models coming from sensors of industrial production lines. This approach uses a graph-based query language (GQL) equivalent to conjunctive queries and has been enriched by inference rules. In both cases, even though semantic approach proves beneficial, they undermine the existing ways of managing complex data and databases at a large industrial scale. The use of a single semantic definition for DT is also not well explored.

Looking at the DT data management side, Zhang et al.[6] proposed an approach to design and develop DT of production line based on semantic data model as a reference model and synchronisation of equipment at the physical level. Angrish[7] introduced an architecture based on a database and generic machine access library for the virtualisation of the production factories. Uhlemann et al.[9] proposed a multi-model data acquisition approach to minimise the delay between the time of data acquisition and creation of production process DT. Consistency check of DT data model is demonstrated by Talkhestani[11] within manufacturing systems. Although these approaches suggest significant data management benefits, still many industries use traditional database management tools such as Structured Query Language (SQL). The emergence of GQL is unlocking new horizons for data storage and visualisation. Therefore, there is a need for a way to link ontology and databases for future DT data management. In this paper, the potential use of ontologies and data modelling for future DT data management has been discussed. For a new concept like DT, understanding the context of data is important and what questions should be asked to make sense out of that data. The freshness and completeness of data, merging of structured and unstructured data is still an ongoing challenge for DT. Use of ontology and data modelling can be one of the viable answers.

### DT ontology model

A generic DT ontology model for an asset during its operational phase of the lifecycle has been proposed as illustrated in Figure 3. The classes of the ontology model are inspired by system architecture for the intelligent and predictive maintenance of an asset.[37] The ontology model not only captures domain knowledge and maintains the semantics of asset functions during operation but also inherits the basic characteristics of an asset DT for further simulations and analysis. The ontology model proposed is the generic representation of an asset DT during its operational phase. Model fulfils both the following requirements:

- Domain knowledge of asset behaviour analysis in operation with essential semantics
- Potential representation of a DT for an asset in operation phase

The overall schematics of model is divided into three predefined information flow layers: physical, data and model layer which fulfil the open architecture of DT. Each layer of the model has been assigned with respective classes.

### Methodology

Keeping the current ways of managing data and databases at a large industrial scale, a methodology has been developed that uses ontology model to create and manage future databases for DT.

The overall methodology is as follows:

1. **Map:** Map the classes of proposed ontology model among functional layers of method/process of asset behaviour analysis.
2. **Define:** Define key data elements and their types for each class of the proposed ontology model.
3. **Create:** Create ontology model by converting relations between classes as object properties and inserting data elements as data properties with logical restrictions.
4. **Convert:** Convert ontology model into a relational data model. Apply keys and cardinality.
5. **Populate:** Populate the relational data model with real datasets.

## Case study

### Ontology model and methodology validation

To validate the proposed methodology, a standard case of CBM has been used. There are various international

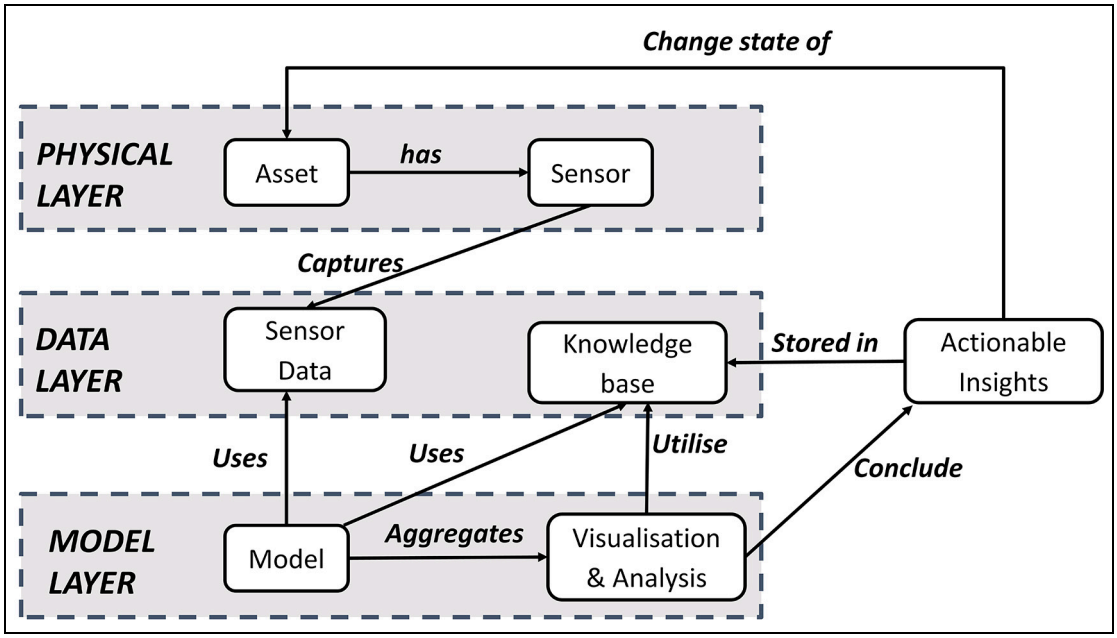


**Figure 3.** Digital twin ontology model.

**Table 1.** OSA-CBM functional layers.[38]

| Functional layers | Function |
|---|---|
| Data acquisition | Access to sensor or transducer data and record |
| Data manipulation | Perform single or/multi-channel transformation and may apply specialised feature extraction algorithms to gathered data |
| State detection | Comparing features against the expected value |
| Health assessment | Determines system's health undergoing degradation by considering health history |
| Prognosis assessment | Displays the current health state of the asset into the future by considering an estimation of the future stage |
| Advisory generation | Gives out the recommendation for maintenance actions and modification of asset |

standards related to CBM approach, e.g. ISO 13374[38] addresses the Open System Architecture for CBM, held by MIMOSA.[39] The OSA-CBM represents formats and methods for communicating, presenting and displaying relevant information and data. At present, OSA-CBM comprises of six functional layers: data acquisition, data manipulation, state detection health assessment, prognosis assessment and advisory generation, to attain a well-constructed system. The description of these functions is shown in Table 1.

**Map:** The process of mapping is driven by the business requirement. The current functions provide the standard approach for CBM of asset. Therefore, mapping of the ontology classes needs to be synchronised with defined functions. In this case, eight classes of the ontology model proposed have been mapped among six functions of OSA-CBM as shown in Figure 4.

- `Sensor,Asset ∈ f(Data Acquisition) // defining semantics for asset and sensor configuration information`
- `Sensor data, Knowledge base, Model ∈ f(Data Manipulation) // defining semantics for data receiving, filtering and conversion to model readable format. Further model algorithm for damage calculations`
- `Knowledge base, Visualisation & Analysis ∈ f(State Detection) // defining semantics for detecting the current state based on knowledge base historical data and enhanced visualisation by the user`
- `Visualisation & Analysis ∈ f(Health Assessment) // defining semantics for assessing the health based on KPIs such as health and diagnostic state`
- `Visualisation & Analysis ∈ f(Prognosis Assessment) // defining semantics for assessing prognostics available to the user`
- `Actionable insights ∈ f(Advisory Generation) // defining semantics for`

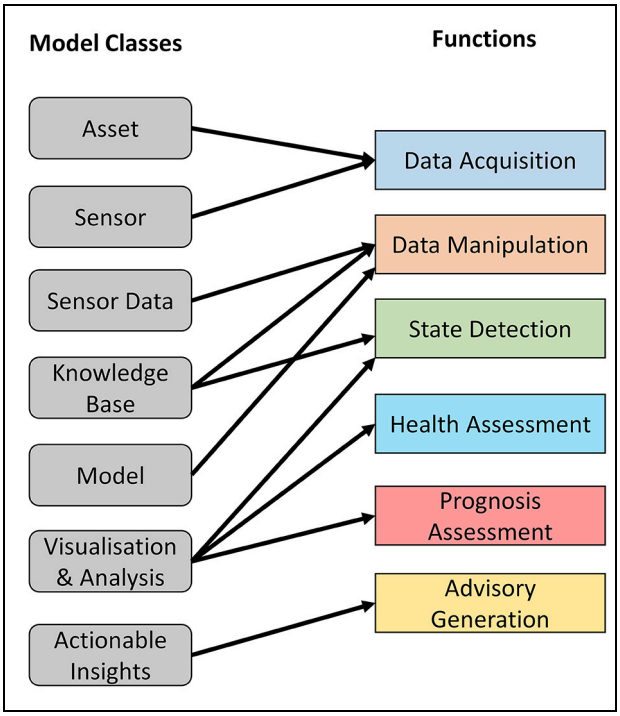


**Figure 4.** Ontology-functional mapping.

`recommended actions essential in operational life.`

## *Define*

To define the data elements for an individual class, an example of aircraft engine display[38] from the standard is used, shown in Figure 5. The example is divided into five distinct areas to provide end-users with a quick summary of the situation. This example is used to derive the potential different data elements and their data types involved irrespective of how data is physically stored or accessed, illustrated in Figure 6. Identification phase describes the configuration aspect of the asset to identify data types such as asset id, report id, etc. Recommendation and prognosis phases provide the idea of different data types associated with suggestions and prognostic results. Health assessment shows the potential data type: health index measure and associated issues identified. State detection phases are the UI/UX interface to understand the current state of asset identifying data types such as vibration amplitude per hours (time).

Figure 6 shows different data elements and their data types that may exist under individual class for aircraft engine CBM example. This declaration is based on the author's assumptions and familiarisation with the domain. The relation between individual classes is denoted, which becomes the object property showing semantic inference.

**Create:** In this step, software Protégé[40] is used for modelling the proposed ontology model. The predefined semantic relationship is converted as object

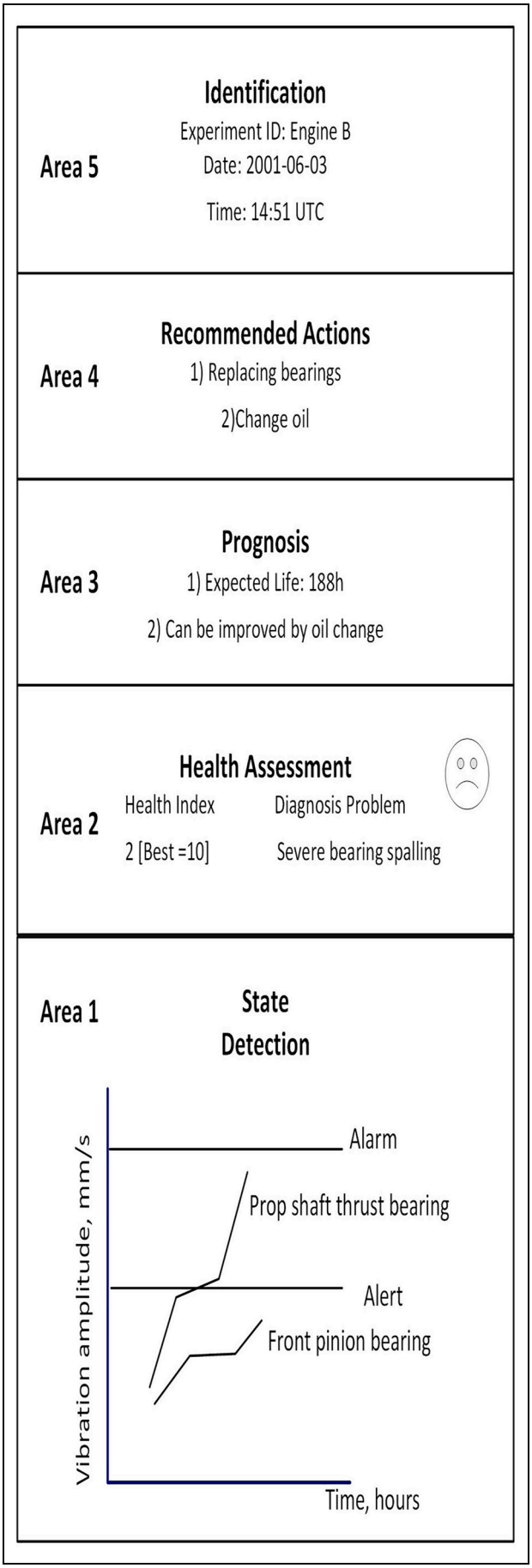


**Figure 5.** Aircraft engine display.[38]

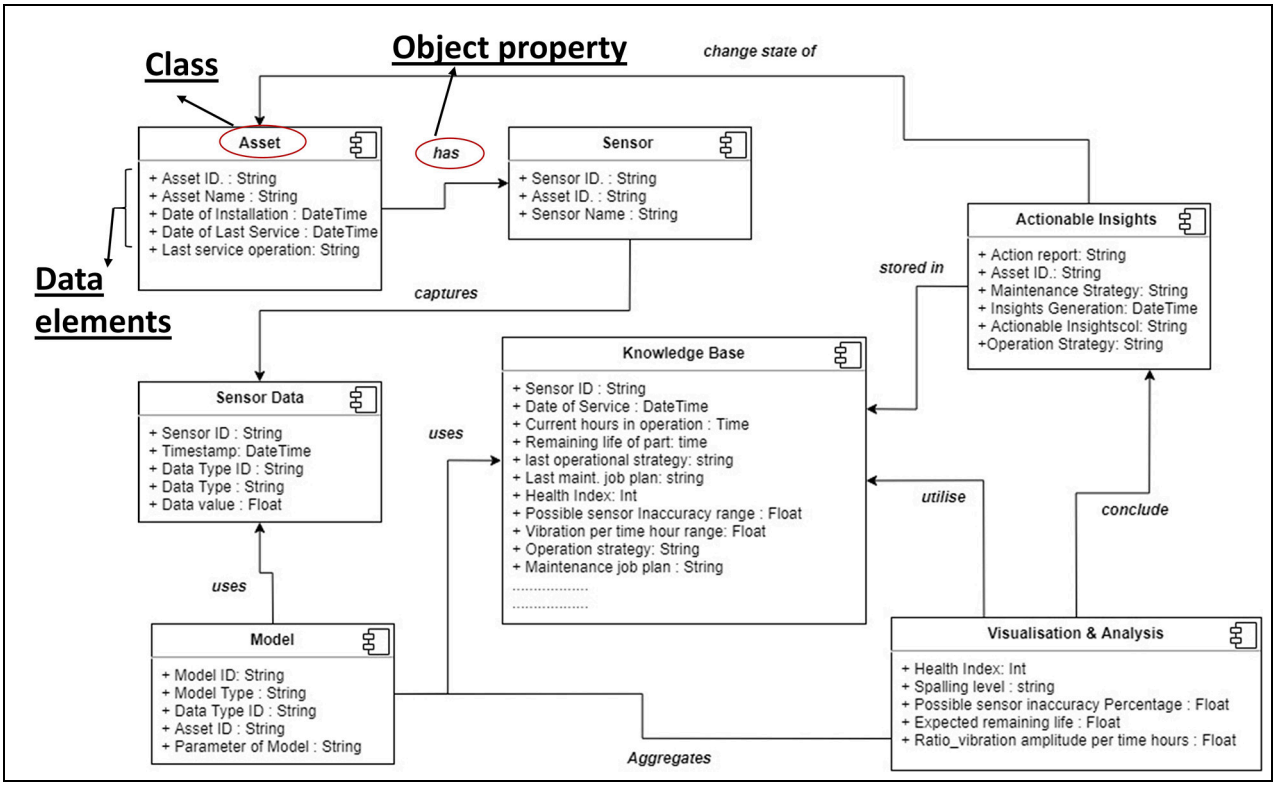


**Figure 6.** Data elements for ontology class.

property in between the classes. Using high-level data elements defined in 'Define' phase, every single data element for each class is defined as data property while creating the ontology model. Each data property is further assigned to a suitable data type with restriction properties. The restriction properties are applied to each object property among two connected classes.

In this step, software Protégé is used for modelling the proposed ontology model. The predefined semantic relationship is converted as object property in between the classes. For example, 'Object property: *has*' is showing semantics between classes 'Asset' and 'Sensor', as illustrated in Figure 7. That means the object property 'has' can be assigned with individual domain 'Asset' and range 'Sensor'.

The restriction properties are applied to each object property among two connected classes. For example, the restriction property defined in between class 'Asset' and sensor with possible cardinality is shown in Figure 8. For class 'Asset', 1 is the minimum cardinality defined for the class- sensor. This means asset has minimum 1 sensor whereas class 'sensor' is restricted with exact cardinality to class 'asset'. This means each sensor identified must belong to an individual asset. Similarly, the restriction properties are established among all the classes of the ontology model. It is clear in the OWL script generated that object property 'has' show the relationship between 'Asset' and 'Sensor' as domain and range

Using data elements defined in 'Define' phase, every single data element for each class is defined as data property while creating ontology model. Each data property is further assigned to a suitable data type with restriction properties. For example, class Asset has data forms: Asset ID, Asset name, Installation_date&time and last_serviced, last service operation has been defined as data properties with related data types. On declaring individual 'A/C Engine' for class 'Asset', the data property assertion is illustrated in Figure 9.

**Convert:** Using the OWL script generated from ontology model, the process of converting ontology into a relational database structure is a step by step process:

**Ontology class to relational data model:** Each class of ontology model is converted into relational database table. The OWL script generated shows declaration of classes in the ontology model. For example, 'Asset' and 'Sensor' are both classes which are converted into relational data model tables.

```
<!--
http://public.cranfield.ac.uk/sxxxxxx/
folder/Test.owl#Asset
-->
<owl:Class rdf:about="http://public.cran-
field.ac.uk/sxxxxxx/folder/Test-.owl#
Asset"/>
<!--
http://public.cranfield.ac.uk/sxxxxxx/
folder/Test.owl#Sensor
-->
<owl:Class rdf:about="http://public.cran-
field.ac.uk/sxxxxxx/folder/Test.owl#
Sensor"/>
```

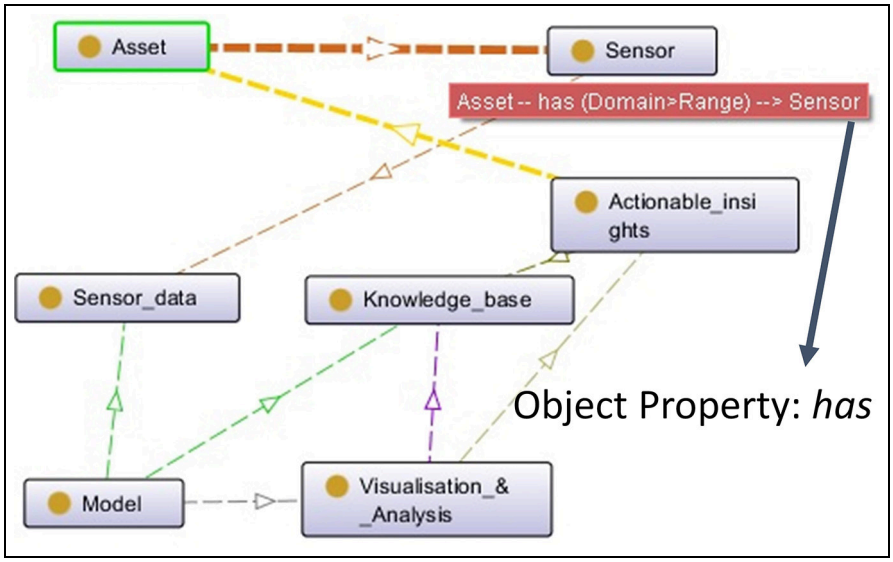


**Figure 7.** Object property between classes.


**Figure 8.** Restriction property.

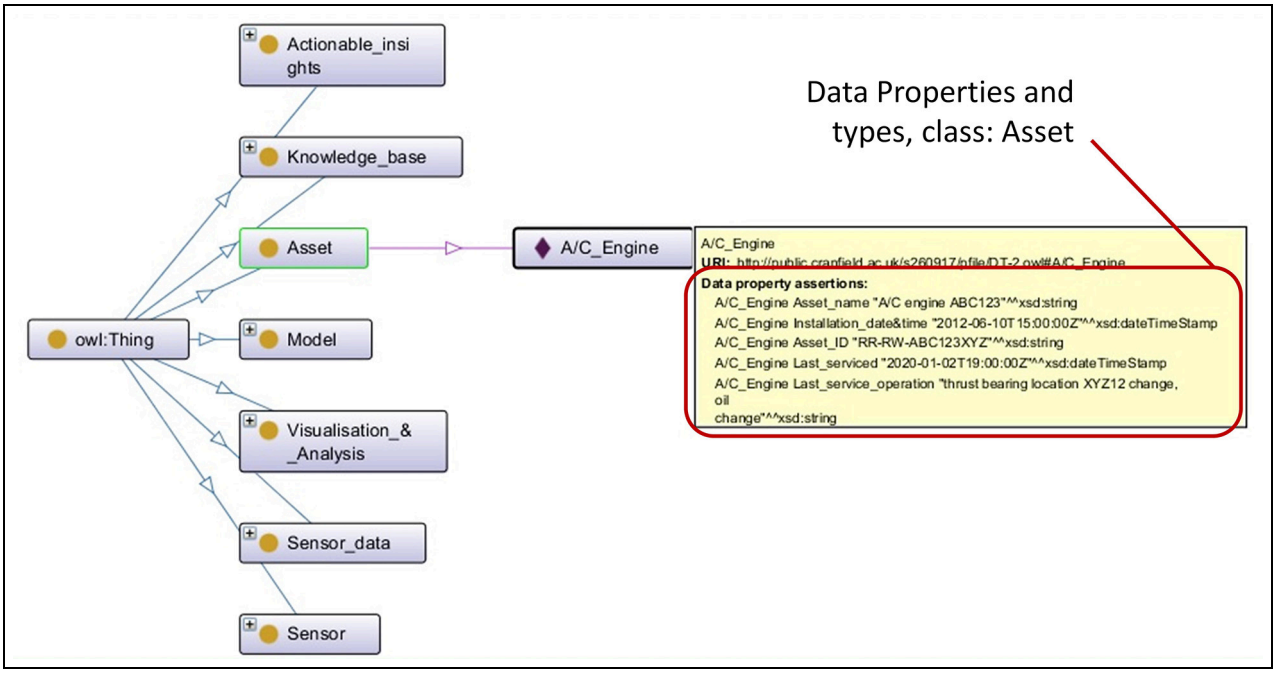


**Figure 9.** Declaration of data properties.

**Ontology Object property to relational data model:** Each object property of the ontology model is converted into relations between data tables. In ontology, when property is defined, it is restricted between the classes with domain and range. Example of OWL syntax, the object property is defined as:

```
owl:ObjectProperty rdf:about="http://pub
lic.cranfield.ac.uk/sxxxxxx/folder/
Test.owl #has" >
 < rdfs:domain rdf:resource="http://public
.cranfield.ac.uk/sxxxxxx/folder/Test.owl
#Asset"/ >
 < rdfs:range rdf:resource="http://public
.cranfield.ac.uk/sxxxxxx/folder/Test.owl
#Sensor"/ >
 < /owl:ObjectProperty >
```

Classes 'Asset' and 'Sensor' are restricted by object property 'has.' Therefore, while transforming ontology to relational data model, object property 'has' is declared as a relation between tables 'Asset' and 'Sensor'.

**Ontology Data Property to relational data model:** For transforming ontology data property to relational database, each data property belong to a single class is declared as data column for that table. Example of OWL syntax, the object property is data property 'Asset_ID' becomes one of the columns for 'Asset' tables of the relational data model. Not only data property, but data type of data property is also transformed into data type for the column. Hence data type 'string' for data property 'Asset_ID' is transformed into data type 'VARCHAR()' for column 'Asset_ID' for table 'Asset.'

```
 < owl:DatatypeProperty rdf:about="http://
public.cranfield.ac.uk/sxxxxxx/folder/
Test.owl#Asset_ID" >
 < rdfs:domain >
 < owl:Restriction >
 < owl:onProperty rdf:resource="http://pub
lic.cranfield.ac.uk/sxxxxxx/folder/
Test.owl #AssetID"/ >
 < owl:allValuesFrom rdf:resource="http://
www.w3.org/2001/XMLSchema#string"/ >
 < /owl:Restriction >
 < /rdfs:domain >
 < /owl:DatatypeProperty >
```

**Ontology Constraint to relational data model:** On transforming the ontology constraints into relational database, formation of metadata tables takes place which becomes the part of the overall data model. An ontology constraint is defined with OWL syntax 'owl:Restriction'. The 'owl:OnProperty' element indicated restricted property. For example, syntax defining a restriction of class property:

```
owl:Class rdf:about="http://public.cran
field.ac.uk/
sxxxxxx/folder/Test.owl #Asset" >
 < rdfs:subClassOf >
 < owl:Restriction >
 < owl:onProperty rdf:resource="http://pub
lic.cranfield.ac.uk/sxxxxxx/folder/
Test.owl #has"/ >
 < owl:minQualifiedCardinality rdf:dataty
pe="http://www.w3.org/2001/XMLSchema
#nonNegativeInteger" > 1 < /
owl:minQualifiedCardinality >
 < owl:onClass rdf:resource="http://pub
lic.cranfield.ac.uk/sxxxxxx/folder/
Test.owl#Sensor"/ >
 < /owl:Restriction >
 < /rdfs:subClassOf >
 < /owl:Class >
```

**Populate:** In the final step, the database is made functional by populating data model with real datasets. This will not only enable developers to understand the data and data types, logic & constraints for constructing databases but also manage the flow of data while DT is in functional mode.

## Query formalisation

For query formalisation, certain sets of assumptions are taken. Health index plays a key role in determining the status of the current and historical state of the asset. Therefore, the classification of health index is assumed among the four categories.

01 – 03: Critical
04 – 06: Poor
07 – 09: Moderate
10: Best

Based on these assumptions, the following query has been formalised:

## Eligibility criteria

**Inclusion criteria:** to be included, an asset must meet the following

1. Has a history of maintenance
2. Has failure in last one month
3. Has current health status ranging between critical to poor

**The Ontology Statement** (as specified by digital twin domain knowledge)

Asset

∃utiliseknowledge base action report id ∩ utiliseknowledgebase heath index ∩ (hashealthindex min 0 ∩ hashealthindex 6) ∩ (hasinsightgenration 2012-06-10T00:00:00Z ∩ hasinsightgeneration 2012-05-10T00:00:Z) ∩ (hashealthindex 0 ∩ hashealthindex 3)

### *Processed ontology statement*

```
Class case study is a sub class of intersection of
  restrcition on property utiliseknowledgebase some value from class :actionreportid
  Interesection of {restriction on property :hashealthindex min 0
  restriction on property :hashealthindex max 6}interesection of
  {restriction on property :hasinsightgeneration min 2012-06-10T00:00:Z restriction on property : hasinsightgeneration min 2012-05-10T00:00:Z} intersection of
  { restriction on property :hashealthindex min 0
  restriction on property :hashealthindex max 3}
  is a sub-class of class:analysis& visualisation
```

## Results

After the transformation of the ontology model, nine main data tables have been obtained for the data model in the relational database MySQL,[41] as shown in Figure 10. The class 'knowledge base' is further simplified among two tables: 'historical data and reports', 'logical data for model'. Table 2 shows metadata values obtained for each restriction property among classes. In total 10 metadata values exist as cardinality restrictions. In the end, this is the minimum database structure that can be used to create and manage data for an asset DT for CBM applications.

The proposed ontology model descriptions are based on DT domain metadata objects which serve as the foundation of handling changes and extension of the system. On query formulation, the domain description is separated from domain metadata from transactional data, thus enabling ease of maintaining semantics to evolve while querying data. The ontology model enables the mapping of classes to data model general schema restrictions (e.g. Null, unique) to restrict data entry. DT domain knowledge is expressed in form of OWL-DL assertions as concept restrictions, which need to be consistent with the respective ontology schemas. Modelling restrictions can be complex and may involve multiple conditions.[25] Existing semantics of the ontology model, the class restrictions are constructed using union, intersection, allValuesfrom, someValuesFrom and complement of OWL-DL ALC (Attributive Language with Complements) constructs. For the query formulation, the quantifier restrictions, someValueFrom and allValuesFrom, and cardinality restrictions are used with object properties and datatype properties etc.

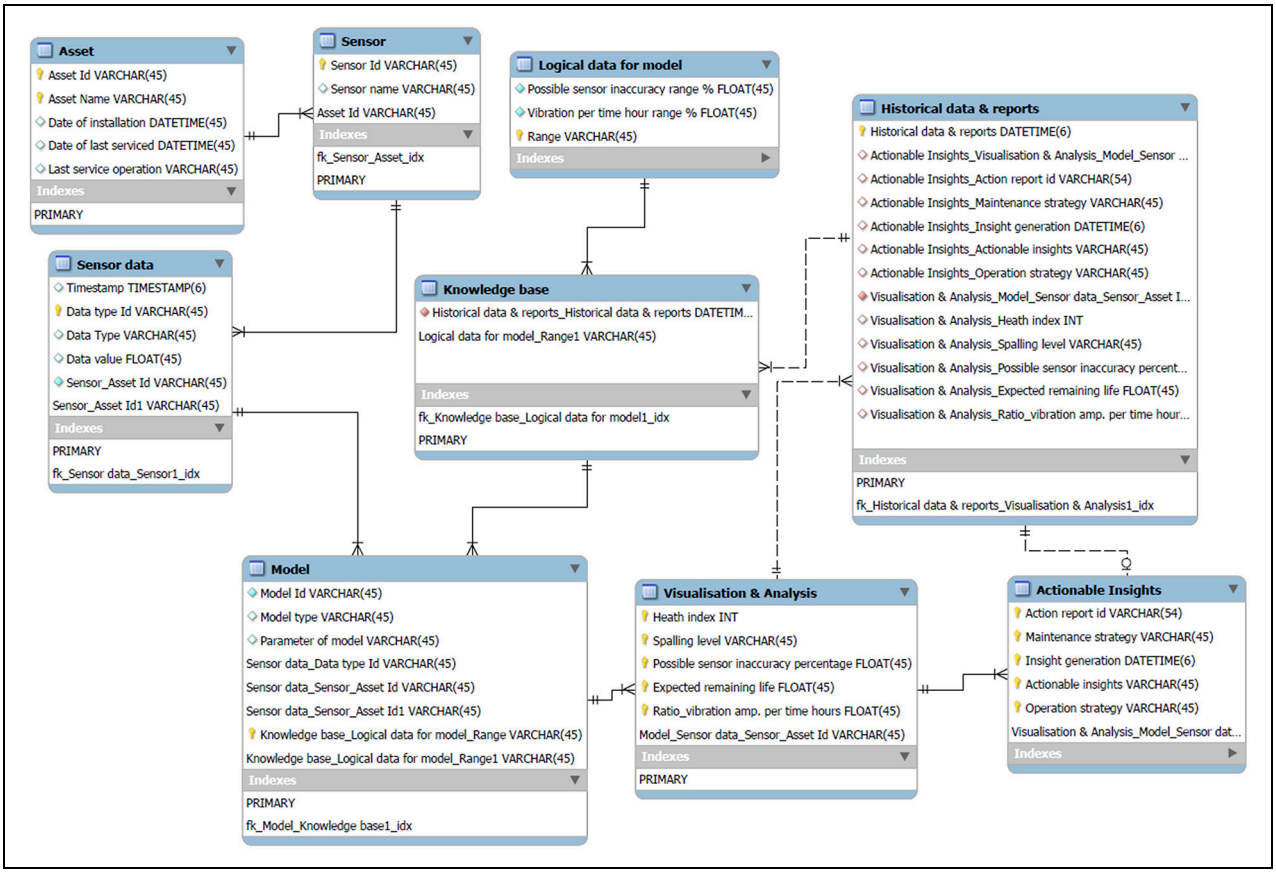


**Figure 10.** DT ontology transformed into relational data model.

**Table 2.** Cardinality metadata table.

| Domain class | Range class | Cardinality | Min cardinality | Max cardinality |
|---|---|---|---|---|
| Asset | Sensor | 1 | 1 | Null |
| Sensor | Sensor data | 1 | 1 | Null |
| Model | Visualisation and analysis | 1 | 1 | 1 |
| Visualisation and analysis | Actionable insights | 1 | 1 | 1 |
| Actionable insights | Knowledge base (historical data and reports) | 1 | 1 | 1 |
| Visualisation and analysis | Knowledge base (historical data and reports) | 1 | 1 | 1 |
| Knowledge base (historical data and reports) | Knowledge base | 1 | 1 | Null |
| Knowledge base (logical data for model) | Knowledge base | 1 | 1 | Null |
| Sensor data | Model | 1 | Null | 1 |
| Knowledge bases (logical data for model) | Model | 1 | Null | 1 |

## Discussion

The data management is a common practice in the development and sustainment of information systems. Effective data management around a particular application or system not only involves data organisation, processing and storage but also include efficient access and retrieval of information to the users. Multi-layer information flow provides the type of important information to be transferred and exchanged between different DT conceptual layers. Transfer of information between the physical layer, data layer and model layer significantly show the essential information important for bringing valuable insights from DT back to the physical asset. Information exchange from model layer to the data layer signifies the minimum information to be retained as historical information which is essential for DT system sustainment. This covers the information retention element for DT which is not well explored in the existing literature. Further, the data model query mechanism explains the user's process of interacting with a DT system by making queries to the DT data model. Query mechanisms highlight the importance and role of a comprehensive DT data model that acts as a gateway between user's requirements, real-time data and historical data.

With the help of the case study, the proposed DT ontology model is used to synthesise the minimum data structure for the DT data model. DT ontology model encapsulates the DT domain ontological concepts for semantic data modelling for DT. The five-step methodology proposed to extract DT data model structure from ontology using ontology-to-conceptual data model transformation. As a result, nine main data tables and 10 metadata values as cardinality restrictions is the minimum data structure for CBM DT application. The minimum data structure for DT data model will play important role in data management right in the initial design phase and development cycle of DT. The methodology provides the freedom to map the existing business functional layers to the ontology classes combining both DT domain and existing business functional knowledge. The minimum data structure will provide exact data definition required for DT of particular scale and application saving expensive and time-consuming efforts of gathering and formalising knowledge for database design. Extraction of minimum data structure will also provide ease of dealing with the problems big data and data complexity simply by providing logical structure for the data model construct. As ontologies are semantically richer than databases, DT ontology model will maintain semantics and concept definition throughout DT lifecycle. This also includes domain rules such as cardinality and disjointness that restricts semantic concept and conceptual relationships in a specific domain.

The quantifier restrictions generated from minimum data structure for DT data model and OWL DL script such as somValuefrom, allValuesFrom and cardinality restrictions will provide the system to deal with DT dynamics challenges (except uncertainty). Continuous update of data within databases is driven by semantic restrictions of DT ontology model proposed. This continuous update of data between different data repositories and IT systems will also aid in maintaining the level of exactness of physical and virtual spaces of DT. The query has been formalised with necessary assumptions which validate the effectivity of query formulation of the proposed methodology. Using the current ontology model, user can query the data without interpretation of transactional data, therefore such data need not be stored as ontology instances. This will help in dealing with the scalability issue of the DT as an interpretation of data sources within existing IT legacy systems and applications can be complex and time-consuming. Thus, the ontology DT ontology model generated plays a significant role in specifying concept restrictions and generating relational database queries.

## Conclusion and future work

Some of the challenges of data management are obvious but virtualisation of the physical asset or product, which is the key validity of DT, brings additional

challenges. In the beginning, questions have been raised in the scope of the DT: *In what ways DT makes the task of data management difficult? What is the minimum level of information required for developing a DT to generate accurate results? How to utilise a knowledge domain concept to drive DT functions? What is most significant data to be used for effective data query and information retrieval?* which became the foundational research questions of the existing work.

At the small scale, DTs can often evolve independently but may need a higher level of data management for the larger-scale application. Industries manage their data with traditional Data Base Management (DBM) systems such as SQL. The question remains that how DTs are implemented, developed and managed in the current data ecosystem. Initially, the data management challenges of DT are identified from the existing literature. DT brings several challenges associated with big data, data volume and variety and issues associated with its dynamics. Such challenges showed the prospects of investigating an effective data management solution which ensures the encapsulation of DT domain knowledge, solves the data structuring complexities for DT and provides the user to build DT data model without having the complete knowledge of the entire domain. In this regard, the information flow between three fundamental layers: physical, data and model layer explained to conceptualise the DT domain knowledge. Based on this knowledge, the DT ontology model is proposed. DT ontology model contains the concept definition, their relationship to domain elements, assertions and domain rules for semantic restrictions. Ontology model not only incorporates the semantics around DT but also helps understand the interdependency of one data on another establishing digital continuity across DT. Ontology model is predicted to be generic enough to map the current ways of managing and analysing an asset during the operational phase.

The five-step methodology, validated with CBM based case study, to extract minimum data structure for DT data model provide advantages of data structuring right in the beginning of DT development cycle, mapping business functions and segregates the most significant data logically structured as the data model. The minimum data structure will provide advantages in terms of DT system scale-up, adding complexity, and continuous update of data along DT lifecycle, by maintaining ontology-driven semantics and domain rules. The mapping of functions to ontology model classes will enhance participation of members that may use DT such as engineers, simulation specialists and teams on the ground. Any change in the ontology model will result in automatic new relationships within the existing data repositories and IT systems. DT ontology model enables the user to run queries without having prior knowledge of the application domain and architectures.

DTs can evolve with time by integrating multidisciplinary DT solutions. The further extension of the existing research is to investigate the efficacy of current concept on the integration of two or more multidisciplinary DTs as a single solution for seamless data modelling and information retrieval. Integration of multiple DTs solutions will not only bring additional complexity to data and information ecosystem but also the opportunity to test the efficiency of knowledge graphs for DT solutions.

## Acknowledgement

The authors would like to thank the Engineering and Physical Sciences Research Council (EPSRC) and AIRBUS for funding this research project. The project is funded via ICASE studentship (2017) which is part of the EPSRC National Productivity Investment Fund.

## Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

## Funding

The author(s) received no financial support for the research, authorship, and/or publication of this article.

## ORCID iD

Sumit Singh https://orcid.org/0000-0001-7991-075X

## References

1. Grieves M and Vickers J. Digital twin: mitigating unpredictable, undesirable emergent behavior in complex systems. In: Kahlen FJ, Flumerfelt S and Alves A (eds) *Transdisciplinary perspectives on complex systems*. Cham: Springer, 2017, pp.85–113.
2. Boschert S and Rosen R. Digital twin—the simulation aspect. In: Bradely D and Hehenberger P (eds) *Mechatronic futures*. Cham: Springer International Publishing, 2016, pp.59–74.
3. Tao F and Zhang M. Digital twin shop-floor: a new shop-floor paradigm towards smart manufacturing. *IEEE Access* 2017; 5: 20418–20427.
4. Parrott A and Warshaw L. Industry 4.0 and the digital twin. *Deloitte Ltd*., https://www2.deloitte.com/insights/us/en/focus/industry-4-0/digital-twin-technology-smart-factory.html#endnote-2 (2017, accessed 1 September 2020).
5. Singh S, Shehab E, Higgins N, et al. Challenges of digital twin in high value manufacturing. SAE technical paper 2018-01-1928, 2018.
6. Zhang H, Liu Q, Chen X, et al. A digital twin-based approach for designing and decoupling of hollow glass production line. *IEEE Access* 2017; 5: 26091–26911.
7. Angrish A, Starly B, Lee YS, et al. A flexible data schema and system architecture for the virtualization of manufacturing machines (VMM). *J Manuf Syst* 2017; 45: 236–247.
8. Yun S, Park JH and Kim WT. Data-centric middleware based digital twin platform for dependable cyber-physical systems. In: *International conference on ubiquitous and future networks (ICUFN)*, Milan, Italy, 4–7 July 2017, pp.922–926. Milan: IEEE.

9. Uhlemann THJ, Schock C, Lehmann C, et al. The digital twin: demonstrating the potential of real time data acquisition in production systems. *Procedia Manuf* 2017; 9: 113–120.
10. Schroeder G, Steinmetz C, Pereira CE, et al. Visualising the digital twin using web services and augmented reality. In: *2016 IEEE 14th international conference on industrial informatics (INDIN)*, Poitiers, France, 19–21 July 2016, pp.522–527. IEEE.
11. Talkhestani BA, Jazdi N, Schlögl W, et al. A concept in synchronization of virtual production system with real factory based on anchor-point method. *Procedia CIRP* 2018; 67: 13–17.
12. Ríos J, Mas F, Oliva M, et al. Framework to support the aircraft digital counterpart concept with an industrial design view. *Int J Agil Syst Manag* 2016; 9: 212–231.
13. Singh S, Shehab E, Higgins N, et al. Towards effective data management for digital twin. In: *Proceedings of the 17th international conference on manufacturing research*, Belfast, 10–12 September 2019, pp.167–172. IOS Press.
14. Vazan P, Janikova D, Tanuska P, et al. Using data mining methods for manufacturing process control. *IFAC PapersOnLine* 2017; 50: 6178–6183.
15. Tyagi P and Demirkan H. Data Lakes: the biggest big data challenges, http://analytics-magazine.org/data-lakes-biggest-big-data-challenges/ (2018, accessed 1 September 2020).
16. Tao F, Zhang M and Cheng J. Digital twin workshop: a new paradigm for future workshop. *Comput Integr Manuf Syst* 2017; 23: 1–9.
17. Wagg DJ, Worden K, Barthorpe RJ, et al. Digital twins: state-of-the-art and future directions for modelling and simulation in engineering dynamics applications. *ASCE-ASME J Risk Uncert Eng Sys Part B Mech Eng* 2020; 6: 3.
18. Kennedy MC and O'Hagan A. Bayesian calibration of computer models. *J R Stat Soc Ser B (Statistical Methodol)* 2001; 63: 1369–7412.
19. Market Research. Digital twin market research report, https://www.marketsandmarkets.com/Market-Reports/digital-twin-market-225269522.html (2020, accessed 1 September 2020).
20. Microsoft Inc. Azure digital twins, https://azure.microsoft.com/en-us/services/digital-twins/ (2020, accessed 1 September 2020).
21. Siemens. Digitalization in industry: twins with potential, https://new.siemens.com/global/en/company/stories/industry/the-digital-twin.html (2020, accessed 1 September 2020).
22. Gilabert E, Conde E and Abaunza J. A standard-based data model for configuration management and maintenance support. *IFAC Proc Vol* 2012; 45: 139–144.
23. Gruber TR and Gruber TR. Translation approach to portable ontology specifications. *Knowl Acquis* 1993; 5: 199–220.
24. Martins CT, Azevedo A, Pinto HS, et al. Towards an ontology mapping process for business process composition. In: Azevedo A (ed.) *Innovation in manufacturing networks. BASYS 2008. IFIP – the international federation for information processing*. Boston, MA: Springer, 2008, pp.169–176.
25. Munir K and Anjum MS. The use of ontologies for effective knowledge modelling and information retrieval. *Appl Comput Informatics* 2018; 14: 116–126.
26. Astrova I, Korda N and Kalja A. Rule-based transformation of SQL relational databases to OWL ontologies. In: *Proceedings of MTSR. Springer*, 2007, pp.1–16.
27. El-ghalayini H, Odeh M, Mcclatchey R, et al. Reverse engineering ontology to conceptual data models. In: *IASTED international conference on databases and applications, part of the 23rd multi-conference on applied informatics*, Innsbruck, Austria, 1416 February 2005, pp.222–227. ASTED/ACTA Press.
28. Vysniauskas E and Nemuraite L. Transforming ontology representation from OWL to relational database. *Technol Control* 2006; 35: 333–343.
29. Gali A, Chen CX, Claypool KT, et al. From ontology to relational databases. In: Wang S, Tanaka K, Zhou S, et al. (eds) *Conceptual modeling for advanced application domains. ER 2004. Lecture Notes in Computer Science*, vol. 3289, Berlin, Heidelberg: Springer, 2004, pp.278–289.
30. Sun C. *A natural language interface for querying graph databases*. MIT, USA, https://dspace.mit.edu/bitstream/handle/1721.1/119708/1078222310-MIT.pdf?sequence = 1 (2018, accessed 1 September 2020).
31. Vicknair C, Nan X, Chen Y, et al. A comparison of a graph database and a relational database: a data provenance perspective. In: *ACM SE '10: Proceedings of the 48th annual southeast regional conference*, Oxford, MS, 15 April 2010, pp.1–6. New York, NY: ACM.
32. Miller JJ. Graph database applications and concepts with Neo4j. In: *Proceedings of the SAIS conference*, Atlanta, GA, 23 March 2013, vol. 2324, no. 36, pp.141–147. AIS eLibrary.
33. Erkoyuncu JA, del Amo IF, Ariansyah D, et al. A design framework for adaptive digital twins. *CIRP Ann Manuf Technol* 2020; 69: 145–148.
34. Bao Q, Zhao G, Yu Y, et al. Ontology-based modeling of part digital twin oriented to assembly. *Proc IMechE, Part B: J Engineering Manufacture* 2020; 1–13.
35. Mehdi G, Roshchin M and Runkler T. Internet of Turbines: an outlook on smart diagnostics. In: *Annual conference of the prognostics and health management society 2017*, St. Petersburg, FL, 2017, pp.1–7. The Prognostics and Health Management Society.
36. Banerjee A, Mittal S, Dalal R, et al. Generating digital twin models using knowledge graphs for industrial production lines. In: *9th international ACM web science conference 2017*, Troy, NY, 25 June 2017.
37. Cachada A, Barbosa J, Leitão P, et al. Maintenance 4.0: intelligent and predictive maintenance system architecture. In: *2018 IEEE 23rd international conference on emerging technologies and factory automation (ETFA)*, Turin, Italy, 4–7 September 2018, pp.139–146. IEEE.
38. British Standard. BS ISO13374-1:2003- Condition monitoring and diagnostics of machines—data processing, communication and presentation—Part 1.1.
39. mimosa.org. Mimosa-an operations and Maintenance information open system alliance, 'MIMOSA OSA-CBM', http://www.mimosa.org/mimosa-osa-cbm/ (2010, accessed 1 September 2020).
40. Stanford University. Protégé, https://protege.stanford.edu/ (2020, accessed 1 September 2020).
41. Oracle Corporation. MySQL, https://www.mysql.com/ (2020, accessed 1 September 2020).