Proceedings of the Sixth International Conference on
Railway Technology: Research, Development and Maintenance
Edited by: J. Pombo
Civil-Comp Conferences, Volume 7, Paper 8.14
Civil-Comp Press, Edinburgh, United Kingdom, 2024
ISSN: 2753-3239, doi: 10.4203/ccc.7.8.14

A Generic and Original Data Model Based on Real World Return of Experience to Support Innovative Tools for Predictive Maintenance

S. Chaumette[1,2] and J. Ouoba[2]

[1]Laboratoire Bordelais de Recherche en Informatique (LaBRI), UMR 5800 (University of Bordeaux, CNRS, Bordeaux INP), Talence, France
[2]R&D, Preditic, Bordeaux, France

Abstract

Maintenance has become a critical issue across all sectors of industry. It is not only a technical necessity but also a major, if not the primary, cost driver. With the advancement of technology, maintenance processes are increasingly driven by data collection. Vast amounts of data are gathered, stored, and analysed to provide valuable management insights. This data comes in various formats and encompasses multiple relationships, making proper structuring essential. The data structure must thus be as open as possible, meaning it should be extensible and capable of integrating new data types. Drawing from the feedback and experience gained through our extensive work with the major French railway companies, we have developed a generic and innovative data model. This model supports a cutting-edge tool we call the Intelligent Digital Data Twin (IDDT, or Preditic-IDDT in reference to our company's name) for predictive and preventive maintenance. Unlike traditional Digital Twins, Preditic-IDDT focuses on the data flow surrounding the system under study rather than the physical system itself.

Keywords: data, data model, open platform, digital twin, railway infrastructure, ETL, condition-based maintenance, predictive maintenance, preventive maintenance.

1 Introduction

Maintenance has become a prominent issue in the railway domain. It is not only a technical necessity but also one of the major cost items, and it significantly impacts the perception of the service by the clients. Until a few years ago, maintenance was conducted based on human inspection in the field. Today, the process has shifted to data collection. Tremendous amounts of data are collected, stored, and analysed to provide operators with proper insights.

Regardless of the industrial installation—whether a railway network, rolling stock, or a factory—it has become increasingly essential to have a virtual (*i.e.*, software) version of the system to manage. These software systems are called Digital Twins[1]. These twins closely replicate the physical aspects of the target system, enabling effective monitoring and control of its physical behaviour.

However, we believe that building a comprehensive vision of an entire system, including its operational environment, involves more than just mimicking its physical attributes. It requires representing all the existing data flows. With this in mind, PREDITIC [2] initiated a R&D program in early 2021 centred around the concept of an Intelligent Digital Data Twin[2], which encompasses these data flows. PREDITIC[3] is an innovative company specialising in the collection, utilisation, and enhancement of business data in the industrial and railway sectors through a modular and scalable data platform. From data collection to the implementation of condition-based maintenance (using Machine Learning), PREDITIC distinguishes itself through its expertise in data analysis in complex environments.

Of course the ETL (Extract, Transform, and Load) [4] process is central to this Intelligent Digital Data Twin. With the advancement of hardware and the variety of data collection systems available today (*e.g.*, handwritten records, sensors, USB keys, connected or over-the-air methods, etc.), Preditic-IDDT is designed to receive and process data from a wide range of sources.The collected data comes in various formats and embeds important relationships. To ensure efficient processing and analysis, it is thus essential to organise this data and these relationships properly. The resulting structure must be as open as possible, meaning it should be extensible and capable of integrating new data types and new relationships. Based on the feedback and experience we have gathered from our extensive work with the French railway companies (among which SNCF and MESEA, a subsidiary of Vinci), we have developed a generic and original data model. Thanks to this model Preditic-IDDT supports innovative tools for condition-based (predictive and preventive) maintenance.

This paper is organised as follows. First, we describe the goals of our work and the methods used to design and implement the data model of Preditic-IDDT. Next, we provide a detailed description of this high-level data model. We then discuss its impact

at the software level. Finally, we conclude with an outline of future work and research directions.

2 Goals and methodology

Our advocated approach is based on the widely accepted hypothesis—although not always implemented as such—that a digital twin system cannot be monolithic. It must be flexible enough to accommodate varied data types and diverse storage systems. The uncertainties we identified in existing systems include the completeness of coverage of data types, the completeness of coverage of data sources, and the ability to combine or assemble different architectures (such as storage systems and servers).

Based on our work with customers, we then have formalised a generic and extensible data model that integrates all the entities essential to Preditic-IDDT functionalities. This model enables the structuring of data for analytical and computational purposes, including the generation of indicators. Additionally, we have instantiated the data model to accommodate various types of storage systems in the platform's initial version. To our knowledge, few, if any, other architectures offer the level of flexibility provided by our system. This openness represents the primary challenge and contribution of the data model that we present in this paper.

Our methodology is grounded in the insights we have gained through an experimental approach and the feedback from various products we have already developed. Utilising this knowledge, we created prototypes and defined demonstration scenarios to showcase the platform's functionalities. Additionally, we are advancing this approach within the framework of the Ferrocampus cluster. "*A unique project in France, Ferrocampus is the European centre of excellence dedicated to rail. This centre of expertise is both a training, innovation and technology transfer hub, based in Saintes, in Charente-Maritime.*"[5]

Our R&D efforts are focused on incrementally developing a platform through successive prototypes, from which we gather comprehensive feedback. This platform is not merely an integration of new modules; the concepts embedded within it permeate and influence its entire design and development, contributing to its uniqueness.

For instance, the platform must be versatile enough to accommodate any type of sensor and data source (*e.g.*, client IT systems, external databases, feedback from field workers, sensors, etc.). It must support all communication protocols, function effectively with sporadic or even absent network connectivity, ensure security, handle large quantities of data, perform advanced analyses, and anticipate future developments (including predictive analytics and AI/ML, such as Deep Learning).

To our knowledge, there is no comparable solution on the market that offers such a holistic and versatile approach.

3 Definition of a generic, high level and open data model

The data model that we have defined and implemented is illustrated in Figure 1. The various components of this model are described in the current section.

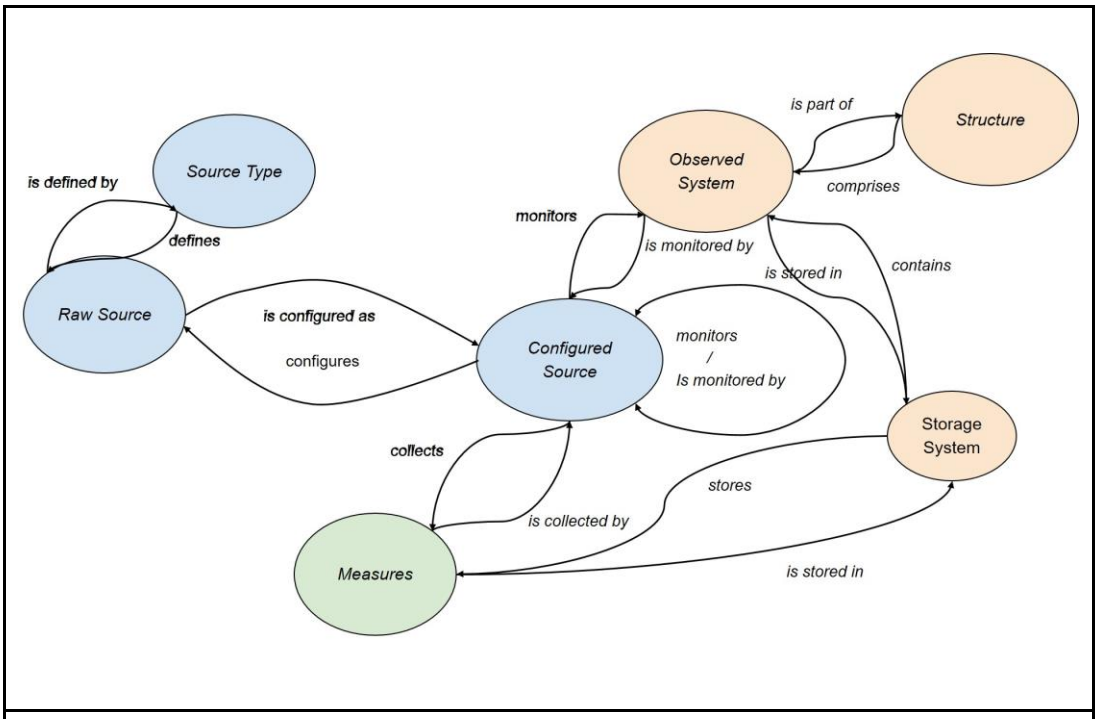

Figure 1: Entities and relationships in Preditic-IDDT

High level entities

We initially adopted a high-level approach to identify all the entities involved in the system to be monitored and those participating in the monitoring process. At this level, collecting data for predictive maintenance involves observing the state of physical systems, referred to here as Observed Systems, by gathering various data about the systems and their environments from multiple sources. Raw data Sources (*e.g.*, a pressure sensor) are strategically placed and calibrated relative to the Observed Systems, transforming them into Configured Sources (*e.g.*, a calibrated pressure sensor) that provide accurate measurements of the physical systems or their environments.

Moreover, when monitoring certain critical systems, the accuracy of the measurements obtained from the Configured Sources is of utmost importance.

Therefore, it may also be necessary to monitor the Configured Sources themselves to ensure they are not faulty. This capability is integrated into the model.

We also consider that an organisation (*e.g.*, a company) may wish to monitor one or more Observed Systems. To accommodate this, we then group these Observed Systems under an entity called Structure. The term Structure has been chosen as it is less specific than organisation.

From a practical standpoint, we have defined the following entities to encompass all aspects involved in the maintenance data collection process described above:

- Observed System: a physical system from which data is collected for study. At the implementation level, an ObservedSystem object includes an associated description.

  Example: *Engine of Eiffel Tower East elevator*.

- Raw Source: an entity that supplies data (*e.g.*, sensor, person, or information system). At the implementation level, a RawSource object includes a source identifier, its type, and a description.

  Example: *RPM counter for Engine of Eiffel Tower East elevator* categorised as a *RPM counter*.

- Configured Source: a Raw Source configured, based on the Observed System considered, to facilitate the collection of relevant data pertaining to it. A Raw Source can have multiple configurations, some active and others inactive (kept for historical purposes). A Configured Source contains the configuration of a Raw Source. Each Configured Source is associated with a list of monitored Observed Systems and/or a list of monitored Raw Sources and/or a list of monitored Configured Sources themselves.

  Example: A RPM counter configured to collect data at a given frequency.

- *SourceParameters*: the parameters used by one or more configured sources *(ConfiguredSource*). The structure of this entity depends on the list of parameters of the associated configured sources.

  For example, the parameters {*RPM (revolutions per minutes) sampling frequency, power consumption sampling frequency*} for source *RPM counter for Engine of Eiffel Tower East elevator* are stored in a *SourceParameters* object. This object contains the configuration for all sources using the same parameters {*RPM (revolutions per minutes) sampling frequency*, *power consumption sampling frequency*}.

- Structure: an entity that aggregates all systems (Observed Systems) to monitor within a structure (*e.g.*, a client).

  Exemple : *EiffelTower company* willing to monitor all its elevators.

Relationships between high level entities

We then identified the relationships between the various objects defined in the preceding section. The major ones are depicted in Figure 1.

Types of data

Several types of data are collected and utilised in Preditic-IDDT. This data can be categorised into two main types: media data and tabular data. Media data comprises images, videos, audio, and similar formats, whereas tabular data encompasses any type of structured data stored in files (*e.g.*, CSV or TXT).

Currently, tabular data is the most frequently collected and predominant type in our various database instances. This category includes numerical, categorical, and textual data.

Effective data, data access and data storage

All information related to the collected data that is necessary for monitoring is stored in various objects. These objects may contain details about the origin of the data, the context in which it was collected, and the effective measurements obtained for monitoring purposes. The uniqueness of our architecture lies in our comprehensive approach to data handling. Not only do we collect and clean data, but we also retain all information regarding these processes, such as the methods used for cleaning and how missing data was managed. By doing so, we provide the final user with greater flexibility in managing their measurements, ensuring that no information is lost at any stage.

From a technical standpoint, we chose the JSON format to structure this data.

We propose organising this information as follows (see Figure 1):

- *StorageSystem*: the storage spaces and associated access information (not including credentials, see below) used by one or more structures *(Structure)* to store the data model and measurements.

  For example, 'Elasticsearch' with URL 'xxx' is used to store the data for a given *ObservedSystem* A, while 'PostgreSQL' with URL 'yyy' is used to store the measurements reported for a given *ObservedSystem* B.

- *Credential*: the connection information related to a *StorageSystem* for a given structure.

  For example, ‘Credential ES A’ contains the credentials required for connecting to the Elasticsearch *StorageSystem* of *Structure* A.

- *SourceType*: includes a predefined category of sources along with a description.

- *Measures*: objects that store data collected by a configured source (*ConfiguredSource*) to monitor an observed system (*ObservedSystem*), a raw source (*RawSource*), or a configured source (*ConfiguredSource*)

  For example, the *RPM (revolutions per minutes)* of the observed system (*ObservedSystem) Engine of Eiffel Tower East elevator* are monitored by a RPM counter configured to collect data at a given frequency.

- *DataStatus*: the data status associated with a measure object (*Measures*) to indicate the validity or status of the reported data.

  For example, the *RPM (revolutions per minutes)* recorded at time T are confirmed as valid through a series of validation methods.

- *Reconstructed*: when data is reconstructed (whatever the method), it is stored in a Reconstructed object. Each *Measures* object has an associated Reconstructed object where reconstructed, estimated, inferred, etc., data can be stored.

4 Impact of the data model at the software level

The high-level open data model we defined has, as intended, a significant positive impact on the Preditic-IDDT software platform built upon it. In this section, we demonstrate its positive effects on the openness of the data collection process and on the completeness of the information accompanying the collected data.

Openness of data collection

The aim of the data collection operation is to enable access to any data source (sensor, external information system, etc.), addressing issues of connection and authentication, and to transfer this data to a server that is part of the Preditic-IDDT platform. This process involves managing the security aspects of the communication. Security is ensured by identifying and restricting entry points to the platform (gateways and application layer access), and by securing the associated data flows through encrypted

communications and individualised access management. Furthermore, interaction with data servers, where data is stored, is authorised only for modules running within the system. All necessary information is integrated into the data model, making the process simpler, more open, and extendable.

Based on these elements, we have developed a generic interface in the form of an API that enables sensors or Information Systems (IS) to transmit data (primarily measurements) to our platform. Additionally, we have designed versatile modules to function as gateways between IoT systems (utilising protocols like MQTT and OPC-UA) and our platform, or to retrieve data from REST API endpoints. The data model supports this process by accommodating both raw sources (*RawSource*) and configured sources (*ConfiguredSource*).

To automate the implementation of IoT gateways, particularly for MQTT and OPC-UA protocols, we initially formalised the parameters essential for configuring a gateway. Subsequently, we developed modules that interpret these parameters and autonomously execute the necessary tasks in the background. Alongside fields detailing information for accessing the MQTT broker or OPC-UA server, the formalised parameters include the observed_system_id field, which identifies the ObservedSystem, and the configured_source_id field (*ConfiguredSource*), corresponding to the configuration identifier used when inserting data into the target storage.

Thanks to the data model we have defined, the resulting platform offers a high level of openness in terms of data sources and storage.

Data cleaning and preservation of all associated information

The objective of data cleaning is to enhance data quality by identifying and eliminating outliers, as well as validating data through the calculation of appropriate indicators.

We have established a structure, *DataStatus*, to track changes made to data. In practical terms, when a data anomaly is identified, a *DataStatus* element is created that references both the original data and a status indicating the type of anomaly detected.

Data cleansing is conducted through three sequential phases: (i) Pre-conformity verification where syntax and completeness checks are performed on the data; (ii) Conformity verification where values are checked against one or more fixed references; (iii) Validity Verification where data is validated against historical records. During insertion in the system, a status is assigned to each data block based on the outcome of these checks.

The pre-conformity verification phase involves validating data by its defined type (integer, string, real, etc.), which must align with the type specified during the creation/configuration of the data source (*ConfiguredSource*). Any missing data or data not matching the expected type is rejected. Invalid data is handled as follows: a DataStatus of 1 is assigned to the entire corresponding data block and variables within the data block that fail this validation rule receive a status other than 0 (status 1 indicates a type mismatch, while status 2 signifies missing values).

The conformity verification phase checks data against predefined value intervals. If numerical data falls outside these intervals (set during the configuration of the associated data source), it suggests an error in data retrieval. In such cases the entire data block is assigned a status of 1 and the variable that fails to meet the interval criteria is labelled with a status indicating it is out of range (status 3).

Thanks to the data model that we have defined, all pertinent information regarding data—whether correct, invalid, missing, etc.—is retained within the system along with appropriate descriptions. This capability enables us to advance analysis beyond the capabilities of most systems, which typically discard incomplete or erroneous data.

Data Analysis and Predictive Models

Once the data has been collected, cleaned, and prepared, the analysis phase can begin, which includes building predictive models. The data model described above, along with the associated APIs, greatly simplifies this work. Additionally, no information is lost in the process, even if a piece of data is incorrect or missing. The Preditic-IDDT platform takes full advantage of these features. However, the detailed process of effective analysis and prediction is beyond the scope of this paper.

5 Conclusions and Future Work

At PREDITIC, we have developed an Intelligent Digital Data Twin, known as Preditic-IDDT. Unlike most Digital Twins, Preditic-IDDT focuses not only on the physical aspects of the system but also on the data flows within the system and its environment. This paper highlights the data model we have designed and implemented, which supports a variety of data inputs and formats, as well as different storage systems. The primary characteristic and innovation of our model is its openness.

In addition to structuring the data and designing the overall architecture for data collection, we have defined the main functionalities of the platform from an end-user perspective and implemented them in demonstrators based on representative use cases (scenarios). We assert that the development and prototyping process must be conducted using an Agile approach, in tandem with the definition of the underlying

model. This ensures that decisions made at one stage do not create significant challenges in subsequent stages. This is the methodology we have followed.

Regarding future work, we will focus on three main areas.

First, the purpose of Preditic-IDDT is to compute indicators using Artificial Intelligence techniques tailored to the objectives of industrial site managers. This involves determining, through an explanatory AI approach, the factors that produced the state or value of a given indicator, or even automatically identifying or defining (through a feedback loop) indicators suited to resolving a specific issue. Therefore, we will address the notion of explanatory AI within Preditic-IDDT.

Second, it appears that most Digital Twins are established retrospectively, after the railway, factory, or other system is already built. While this approach is relevant for existing installations, it is less effective when the question could have been addressed during the initial design phase of the system. In the long term, Preditic-IDDT aims to enable the determination of the impact a new data source would have on the quality of the indicators through simple emulation or simulation, before any deployment, which is inherently costly.

Third, we have tailored Preditic-IDDT into a data analysis tool specifically designed to meet the needs of maintenance professionals in the railway sector, both in terms of the content provided and the mode of use (ensuring the tool is simple to operate within the technical environment where the agents work). Naturally, its application in other industrial domains is entirely feasible, thanks to our open data model. Therefore, in our future work, we will explore the integration of other industrial sectors into Preditic-IDDT.

Acknowledgements

The authors would like to express their gratitude to all the employees of PREDITIC for their dedicated work on the platform. Additionally, they extend their thanks to all our customers who, through the systems we have developed for them, have indirectly contributed to the work presented in this paper.

References

[1] Tao, F., Zhang, H. & Zhang, C. Advancements and challenges of digital twins in industry. *Nat Comput Sci* 4, 169–177 (2024). https://doi.org/10.1038/s43588-024-00603-w

[2] Serge Chaumette, Jonathan Ouoba. Key challenges in building an intelligent data twin: the Preditic retex. *18th International Conference on Intelligent Environments (IE)*, Jun 2022, Biarritz, France. ⟨hal-03703294⟩

[3] The Preditic company http://preditic.com/

[4] SEENIVASAN, Dhamotharan. ETL (extract, transform, load) best practices. *International Journal of Computer Trends and Technology*, 2023, vol. 71, no 1, p. 40-44.
[5] Ferrocampus - European centre of excellence dedicated to rail https://territoires.nouvelle-aquitaine.fr/actualites/ferrocampus-se-structure