Ontology-Guided Knowledge Graph Construction to Support Scheduling in a Train Maintenance Depot

Emmanuel Papadakis[1], Thomas Leo McCluskey[1], Hassna Louadah[2] and Gareth Tucker[2]

[1]School of Computing and Engineering, University of Huddersfield, UK
[2]Institute of Railway Research, University of Huddersfield, UK

Abstract
Rolling stock maintenance is essential to ensure train reliability. Adapting automation to optimise the efficiency of train maintenance procedures is crucial to maintain a smooth operation of rail transport. Information related to the scheduling and planning requirements for train maintenance procedures, however, adhere to task-specific schemas or worse, exists in unstructured formats, which hinders computerised analysis and impedes interoperability. To fill this gap, a knowledge acquisition pipeline is introduced to semantically organise textual information found in semi-structured maintenance instruction manuals. XML transformations, pattern matching and elastic search over industrial vocabularies are utilised to automatically populate a domain-specific ontology and construct a knowledge graph that describes maintenance tasks. Through demonstration, we exemplify how the resulting knowledge source can be accessed and used in Short-Term Scheduling operations.

Keywords
Scheduling, Knowledge Graph, Information Extraction, Rolling Stock Maintenance, Ontology

1. Introduction

With the use of rail transport on the increase, more trains are operating on the network, and train reliability becomes increasingly important to avoid disrupting the service. Additionally, a large proportion of rolling stock life cycle costs is related to the preventive and corrective maintenance processes undertaken in the train maintenance depot (henceforth referred to as depot). Hence there is a need to optimise or at least improve the efficiency of the maintenance processes. Given the abundance of data, it is possible to discover ways to improve maintenance efficiency, cost management or to forecast demands to avoid negative events. Most of the information systems involved, however, are built independently using task-specific schemas, resulting in poor interoperability between component systems [1]. Consequently, integration becomes a labour-intensive work of aligning different databases and changing source code. Furthermore, a great deal of information exists in unstructured formats, such as text, impeding computerised analysis that relies on the content of this textual information. In particular, where these information systems feature scheduling or planning operations, as is the case in

Planning and Ontology Workshop (PLATO), ICAPS 2023, July 09-10, 2023, Prague, CZ
e.papadakis@hud.ac.uk (E. Papadakis); t.l.mccluskey@hud.ac.uk (T. L. McCluskey); h.louadah@hud.ac.uk (H. Louadah); g.j.tucker@hud.ac.uk (G. Tucker)
0000-0001-8669-2420 (E. Papadakis)

train maintenance, automating these operations requires detailed ontological knowledge of often company-specific maintenance instructions. Encoding and engineering the knowledge necessary by hand (such as Planning Domain Definition Language - PDDL), in order to automate and optimise such complex activities, would be extremely inefficient, if at all feasible.

The work reported here has been carried out in the context of a £1.8 million project (referred to as the SRS Project below) to establish a "Smart Rolling Stock Maintenance Research Facility" in the Institute of Railway Research, University of Huddersfield [1]. The aim is to carry out research into the improvement of the efficiency and effectiveness of rolling stock maintenance to meet the challenges of the near future of rail. This has led to our development of a rich digital infrastructure called a *Virtual Depot* which attempts to integrate data and maintenance processes such as planning and scheduling into a common framework, based on ontological engineering. Given the complexity and variability of the underpinning knowledge, our project relies in part on the success of employing knowledge acquisition tools to create structures such as knowledge graphs, which represent and connect up heterogeneous data.

The generic question that this paper seeks to answer is as follows: to construct a rich digital infrastructure (such as a Virtual Depot) for an industrial process (here train maintenance) which requires the ability to reason about activities in this process, such as performing planning and scheduling, requires a detailed and complex set of ontologies. While creating or acquiring generic ontologies that capture the essence of such industrial processes is feasible, the peculiar details of company-created processes and procedures may amount to huge amounts of information. For example, in one train company with which we are partnering, they produce circa 250 vehicle maintenance instruction manuals (VMIs) for their products. How is it possible to overcome this knowledge acquisition bottleneck, and harvest the specific knowledge required to provide rich knowledge structures that can be used in scheduling tasks?

This paper describes the methodology to construct a Knowledge Graph[2] about railway maintenance tasks (henceforth RailMain-KG). Utilising the Resource Description Framework (RDF[3]), the RailMain-KG captures and organises information about entities, attributes and relations within the domain of railway vehicle maintenance enabling the automated scheduling of maintenance activities. Guided by a domain-specific ontology and industrial vocabularies introduced by rolling stock maintenance resource management systems, the authors employ information extraction on semi-structured data sources to encode detailed knowledge about maintenance tasks. The resulting Knowledge Graph captures information about maintenance tasks, parts of the vehicle they are applied to, and their requirements in terms of resources. The authors illustrated the knowledge acquisition process by applying it to a large set of VMIs for a particular train manufacturer, and show how the resulting knowledge can be accessed for and used in Short-Term Scheduling operations. Finally, the extremely important issue of validating the resultant knowledge is discussed.

[1]This work is funded by the European Regional Development Fund (ERDF)
[2]https://www.ibm.com/topics/knowledge-graph
[3]https://www.w3.org/TR/rdf-concepts/

2. Related Work

The benefits of fusing planning and description logic have long been recognised by the community of AI planning and scheduling [2]. A common drive behind the applications of this research area is to draw the benefits of ontological modelling and knowledge sharing. In particular, domain knowledge is represented in a formal language, such as an ontology, which provides an intuitive representation of information that is intrinsically interoperable. This knowledge can then be filtered, packed and translated into planning or scheduling-friendly formulations.

In the domain of business, researchers utilise ontologies to inform workflow generation [3], by encoding planning functions, actions and goals into RDF patterns that are converted into domain PDDL models using RDF transformations. In the domain of service composition, [4] uses ontologies to implement a graphplan-like planner. In the area of Intelligent Transportation Systems, domain knowledge is formalised in order to automate problem-solving and reasoning procedures [5], thus facilitating reusability and transferability of knowledge. In space operations, planning operations are fused with ontological models to allow subject experts to encode domain knowledge; this facilitates automatic creation of PDDL components, which overcomes the barrier of PDDL expertise Bonasso et al.. In an industrial setting, ontologies are used to create fundamental knowledge for Industry 4.0 applications [7] enabling automated creation of domain models and explanations for the generated sequence of actions. Other literature relies on entity and relation extraction from text using neural-based natural language processing (NLP and aims to construct knowledge graphs to be used in planning or scheduling maintenance workflow within various domains. Indicatively, Li et al. converts manuals about flight control maintenance into knowledge graphs to automate the planning of maintenance procedures [8]. Another work semantically organises multiple sources of information about industrial equipment by populating a domain-specific ontology to inform maintenance solutions [9]; in the domain of nuclear energy production, [10] structures information about equipment failure and troubleshooting into a knowledge graph in order to assist the planning of overhaul inspection and fault diagnosis[10].

To the best of our knowledge, there has not been any research on constructing knowledge graphs in the domain of rolling stock maintenance. The work presented in [11] focuses on rolling stock maintenance failure, however not enough technical details are provided to allow a fair comparison. To fill this gap, we introduce a knowledge acquisition pipeline to encode information found in VMIs resulting into a knowledge graph, which can further inform scheduling procedures. Note that many relevant works utilise deep learning approaches over annotated corpora to identify key concepts from the text and built knowledge graphs. Considering the diverse structure of VMIs and the scarcity of annotated text in the domain of train maintenance, this work focuses on solutions that do not require labelled data.

3. RailMain-KG Schema

This work relies on a generic ontology to capture the main domain concepts, roles and relations thereof. The knowledge acquisition pipeline is build on RailMaiin-KG schema, that is, an extended subset of the *Depot Ontology* [1], which is an ontology capturing the knowledge regarding train maintenance, ranging from physical entities such as vehicles and resources to information objects like activities and tasks. RailMain-KG schema is built on the key notions described below that form the classes of the graph and they are interlinked as shown in Figure 1

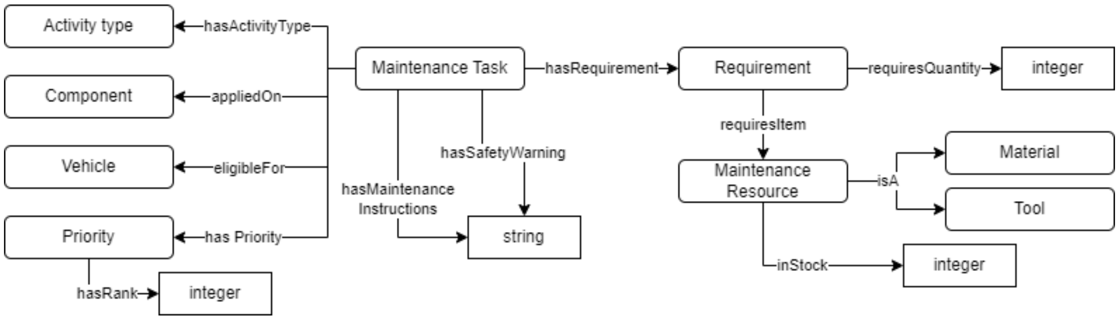

Figure 1: RailMain-KG Schema (Subset of Depot Ontology)

- **Vehicle**: a specific carriage within a train that is subjected to maintenance operations.
- **Maintenance Task**: an information object consisting of activities applied on components of vehicles, in order to ensure their technical integrity.
- **Activity type**: categorisation of maintenance tasks that specify their effect effects on a component, such as inspection, replacement, and cleaning.
- **Component**: systems on a vehicle that deliver a specific function. They can be single elements (such as wheels) or permanently grouped components that form sub-systems and act as single units (i.e., braking system).
- **Requirement**: information objects that express the necessary elements for the successful execution of a maintenance task. They are expressed as pairs of resources coupled with quantity restrictions.
- **Maintenance Resource**: assets in maintenance depots used to carry out maintenance tasks. They are divided into tools and materials.
- **Material**: resources that are consumed during maintenance tasks, for instance, detergents, engine oil as well as machine elements such as screws.
- **Tool**: reusable physical item used to physically manipulate or measure a component, trains system or sub-systems as part of a maintenance task, for example hand-tools, machinery or gauging equipment.
- **Quantity**: a quality that specifies the amount/number of resources needed for a task.
- **Priority**: indication of how critical a maintenance task is; it implies the expert-based estimated risk posed by a component with compromised integrity, which is the result of absent maintenance.
- **Safety warning**: messages describing potential harmful incidents that might occur during a maintenance task.
- **Maintenance Instructions**: a written description of a maintenance task including how to use the necessary tools, materials and replacement components.

4. Building the RaiMain-KG

This section describes the steps to populate the RailMain-KG Schema with detailed knowledge resulting into a knowledge graph (referred as RailMain-KG) about train maintenance procedures. Using the schema as a guide, the graph is formed through knowledge extraction over the train manufacturer's set of rolling stock maintenance instruction manuals, formatted in HTML for online viewing. This procedure converts semi-structured text-based information into a formal representation that adheres to the principles of the Semantic Web and it is organised as a pipeline (Figure 2) consisting of three sub-processes, data acquisition, information extraction and encoding of knowledge as RDF triples.

4.1. Data Acquisition

RailMain-KG is built on rolling stock documentation, in particular vehicle maintenance instructions (known as VMIs). These documents are provided by railway industry manufacturers, and capture guidelines that ensure the mechanical integrity of various components equipped on a vehicle. Although the template of VMIs depends on the manufacturer, their content covers the identification of the task being documented, its essential requirements and a step-by-step procedure.

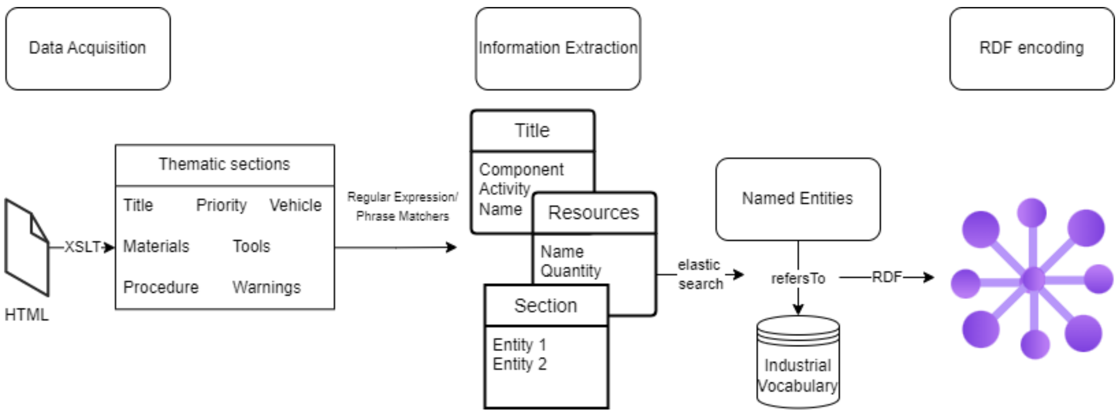

Figure 2: Overview of the knowledge acquisition pipeline

For RailMain-KG, we use VMIs provided by a well-established railway manufacturer, who for the sake of confidentiality must remain anonymous. Each document is formatted as a HyperText Markup Language (HTML) page and provides a thorough description over the guidelines of a single maintenance task applied on a particular component, sub-assembly or vehicle system. The content is encoded as text framed in a semi-structured fashion via the hierarchy introduced by the HTML building blocks and their class attributes. Context-wise, the structure of the VMI documents includes the following sections:

General information is encapsulated in the header of a VMI and includes a title, importance and applicability. Title is formatted as raw text and covers the name of the task and the component it is applied on. Importance is represented as a unique label expressing the criticality of the task according to safety (i.e., normal, critical or high risk), whereas applicability is expressed as comma-separated terms describing the types of vehicle for which the task in question is eligible.

Resource requirements are summarised in two sections, which cover the materials and tools that are essential for the task being described. Each section is provided in tabular format, where each row represents a resource and columns cover its textual description and expected quantity.

Procedure is described as an enumerated list of instructions; occasionally supplementary material may be provided that encompasses safety warnings and other miscellaneous notes.

Although there is a clear separation between the aforementioned sections, their content is in the form of free text, which hinders consistency. Components and the name of the task might appear in any order within the title. Furthermore, cell values within tabular structures are not consistent. For instance, there are references to a single entity using a wide range of spelling variations or interchanging between general descriptions and product brands (such as lubricant spray and WD-40). Inconsistent text structure and terminology raises some serious obstacles in data integration, which we address by employing text mining methods and entity alignment techniques based on deep learning models.

4.2. Information Extraction

The first step in the creation pipeline is to parse a set of VMIs and extract relevant information about tasks that fit the purpose of RailMain-KG. This version of the graph is based on 246 unique VMIs that cover a wide range of tasks ranging from simple inspections and measurements to more advanced activities such as repair or replacement of mechanical parts. During information extraction, each VMI is converted into groups of labelled entities that describe the thematic sections covered within the document. First, we employ several pattern-matching templates implemented in Extensible Stylesheet Language Transformations (XSLT) to organise the content of each VMI into 7 groups that reflect the RailMain-KG schema. These include title, priority, vehicle eligibility, required resources (grouped as materials and tools), safety warnings and procedure. Then, several natural language processing steps transform the textual elements of each group into distinct typed entities that reflect the key notions mentioned in Section 3.

Title of VMIs describes the applicability of a task, for instance, "Engine - Coolant Filter Change". Regular expressions and phrase matchers are utilised to extract a) the activity type of the task, such as change, b) the component, where the task has been applied on (e.g., engine) and c) the actual name of the task.

Information about eligible vehicles, safety warnings and procedures is retrieved through regular expressions that break down comma or new line - separated and enumerated phrases. This step ensures that composite information is expanded into its constituent elements, which in turn are linked to the entities of the RailMain-KG schema. For instance, safety warnings and vehicle types are considered instances of namesake entities. In similar lines, the textual description of the procedure is transformed into a series of indivisible entities that stand for maintenance steps. Information about the criticality of the maintenance task is a self-contained label (i.e., 'mission critical'), which is linked to the entity of significance.

Information extraction is finalised by recognising the essential resources of VMIs along with their specified quantity. Although the quantity of a resource is mined directly from the text, the diverse use of industrial terminology within the sections of tools, materials and protective equipment renders the extraction of the actual resources as an entity-linking challenge. To address this, we convert the tabular structure that summarises the prerequisites of a maintenance task into tuples of resource references and quantities. In order to associate resources to the corresponding entities within RailMain-KG schema, each textual reference is linked to a unique entity drawn from a pool of formal industrial concepts. This collection of terms is suggested by industrial vocabularies, which are constructs primarily used to keep records of the available stock (such as spare parts) within a railway depot. They are formatted as comprehensive catalogues of industrial terms that cover a wide variety of elements that are already classified as materials, tools or equipment used in rolling stock maintenance.

Matching textual references of resources to industrial concepts is accomplished via elastic search over the predefined set of concepts conjured by the industrial vocabularies. This method is implemented with approximate nearest neighbours search based on hierarchical navigable small world graphs [12]. In particular, frequency inverse document frequency (tf-idf) on character 3-grams is used to convert industrial terms into vectors. Projecting terms to a non-metric space enables cosine similarity to be used to retrieve the most relevant industrial concepts given a sample text and a similarity threshold (empirically set to 80%).

4.3. Creation of RDF Triples

The knowledge acquisition pipeline is concluded with the conversion of the extracted information into RDF triples that conform to RailMain-KG schema (Figure 1). Every VMI is an instance of a maintenance task that is populated with several properties that link to entities relevant to the thematic sections introduced in Section 4.2. Simple entities or collections of the same type, such as vehicles, are represented by repeating (as needed) properties. Composite entities such as resources and procedures are implemented using reification. In this case, the task in question is connected with an intermediate object that aggregates the individual properties of the associated entity. For instance, a resource is represented by an object that encapsulates both its type and required quantity. The creation of the graph and its serialisation is achieved using RDFLib library[4]. Figure 3 illustrates the conversion of a task into a graph-like format.

5. Accessing Knowledge in RailMain-KG

During our fieldwork, we have characterised rolling Stock maintenance depots as requiring several distinct scheduling operations depending on the granularity of scheduling objects and overall time span. A three-stage (long, short, daily) collection of scheduling processes seems the most natural structure. Long-term scheduling (horizon is in the order of years) is invariably automated in Maintenance Depots, where abstract schedules of maintenance contain due dates of periodic maintenance of train units, and are arranged according to manufacturers' specifications. In contrast, short-term scheduling processes (horizon is approximately 1-2

[4]https://github.com/RDFLib/rdflib

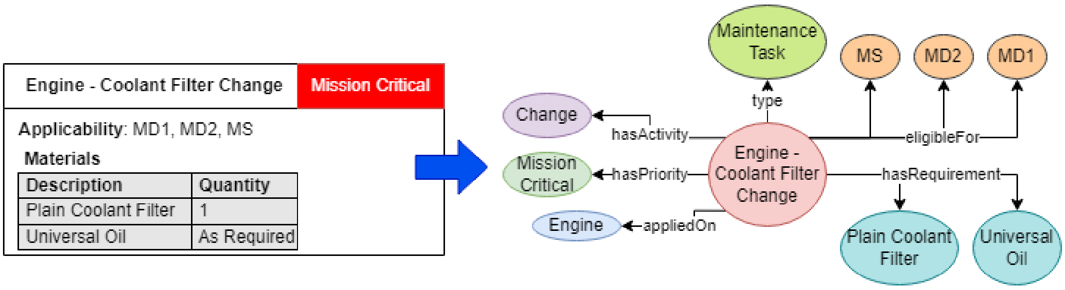

Figure 3: Graph representation of a task for changing coolant filter in the engine. Note: Data properties are skipped for the sake of readability.

weeks) appear to be poorly supported by tools, and it is this area where we have focused. This requires information about the maintenance interventions' due dates (provided by the long-term schedule), fleet availability constraints, day restrictions and workforce in addition to specific resources. Indeed, each maintenance intervention has specific depot resource requirements, including fixed/mobile depot resources, tools, and materials. Automating short-term scheduling requires up-to-date information about the availability of the required resources. However, as of today, the tools and materials required for each maintenance intervention is mainly stored in VMI documents and technicians' heads. The construction of the RailMain-KG above is a major step in the full automation of the scheduling process (a description of the automation of a short-term scheduling process in this train maintenance domain is given in [13]). This section describes how scheduling operations and other processes can access the knowledge harvested in RailMain-KG, through examples. In particular, we demonstrate how, since the results of the semantic search are generated into a digitized format, they may be accessed and used directly by scheduling scripts. The results are generated by issuing a SPARQL query against an instance of RailMain-KG using the GraphDB[5] platform and visualised accordingly.

Resource-aware scheduling: A common strategy for resource management that maintenance depots adopt is a recurrent resupply cycle. Spare parts and other materials are being replenished on the basis of fixed thresholds depending on the available stock. This method ensures that there is always a surplus of resources regardless of the actual demand. Despite the effectiveness of this solution, it does not harbour any intelligence, which impedes automation. For instance, changes in depot policies, such as the introduction of more frequent maintenance tests or cancelling others may lead to resource inadequacy or excessive stock. Information within RailMain-KG can be accessed by the short-term scheduling process to account for the availability of resources in advance, hence skipping tasks when resources are not enough, prioritising critical tasks based on the available stock, and allowing active management of stock level to ensure all required maintenance tasks can be carried out on time. Hence, the proposed approach contributes to the spare parts management and preventive maintenance scheduling processes: rolling stock maintenance costs have around 27 influential variables [14]; the highest among the influential costs are 13.8% spare part cost, 11% life cycle cost, 6.4% preventive maintenance costs. Figure 4 depicts a query that aggregates the requirements of a set of maintenance

[5]https://www.ontotext.com/products/graphdb/

tasks, commonly denoted as exams. For the sake of simplicity, the exemplified exam contains 7 random tasks; the result is a collective overview of the necessary material complemented with the stock availability (in-stock quantities are arbitrarily introduced for demonstration purposes).

```
select distinct ?material ?inStock ?quantifier
(IF(MIN(?quantity) = 0, SUM(?quantity)+COUNT(?quantity), SUM(?quantity)) AS ?required)
where {
   values ?t {srsm:Task1 srsm:Task2 srsm:Task4 srsm:Task0} # random tasks
   ?t srsm_s:hasRequirement ?req.
   ?req srsm_s:requiresItem ?material; srsm_s:requiresQuantity ?quantity.
   ?material rdf:type srsm_s:Material;
          rdfs:label ?item; srsm_s:inStock ?inStock.
   BIND (IF(?quantity = 0, "at least","exactly") AS ?quantifier)
} group by ?material ?item ?quantifier ?inStock limit 5
```

	material	inStock	quantifier	required
1	srsm:M31	"9"^^xsd:integer	"exactly"	"16"^^xsd:integer
2	srsm:M1587	"1"^^xsd:integer	"exactly"	"2"^^xsd:integer
3	srsm:M8	"5"^^xsd:integer	"at least"	"2"^^xsd:integer
4	srsm:M37	"7"^^xsd:integer	"at least"	"1"^^xsd:integer
5	srsm:M76	"2"^^xsd:integer	"at least"	"2"^^xsd:integer

Figure 4: Aggregation of requirements

Filter Tasks by resources: Due to unforeseen circumstances, a supplier announces a recall for the last batch of engine oils. This event causes major changes in the current schedule because it cancels out maintenance tasks that depend on the affected material. Currently, spotting problematic tasks relies on expertise or requires manual scanning of the instruction manual of every scheduled task. RailMain-KG automatically addresses this question by retrieving tasks from a predefined collection, which considers engine oil as a prerequisite. The maintenance scheduling process may take this information to delay the scheduling of tasks requiring engine oil until it is available. Figure 5 illustrates a query that retrieves all the maintenance tasks that are applied on a vehicle's engine and require engine oil. The result indicates the "Engine - Oil Level Check" is the only affected task. Reversing the previous query, it is possible to generate recommendations of alternative tasks which can be used as input for new scheduling procedures or be used to replace the problematic tasks.

	task	appliedOn	reqlabel
1	"Engine - Oil Level Check"	"Engine"	"15w40 multiviscosity engine oil"

```
select ?task ?appliedOn ?reqlabel
where {
    ?t rdf:type srsm_s:MaintenanceTask; rdfs:label ?task;
        srsm_s:hasRequirement ?req; srsm_s:appliedOn srsm:C11. #Engine
    srsm:C11 rdfs:label ?appliedOn.

    ?req srsm_s:requiresItem srsm:M99. # engine oil
    srsm:M99 rdfs:label ?reqlabel.
} group by ?task ?appliedOn ?reqlabel
```

Figure 5: Semantic search: Find all tasks applied on the Engine that use engine oil.

Task Priority: Maintenance supervisors are responsible for determining the importance of maintenance tasks (discovered during inspections or received as part of the daily reports containing all the raised issues that require the supervisor's input) based on their effect on vehicle parts. For instance, inspecting the integrity of wheels is considered a high-risk task, which, if skipped, might render the vehicle incapable of operation. Supervisors conform to the guidelines of the manufacturers and current maintenance authority to inform how critical some tasks are before they are subjected to scheduling. To this end, RailMain-KG can provide customised decision support by prioritising maintenance tasks. Figure 6 illustrates a query that orders and groups a predefined set of tasks based on their estimated risk. Applying additional filters enables further customisation of the results, such as retrieving critical tasks that are applied on the air system.

```
select distinct ?task ?priority ?component
where {# random tasks
   values ?t {srsm:Task0 srsm:Task1 srsm:Task2 srsm:Task189}
   ?t srsm_s:hasPriority ?p;
      srsm_s:appliedOn ?c; rdfs:label ?task.
   # rank is determined by the risk
   ?p rdfs:label ?priority; srsm_s:hasRank ?rank.
   ?c rdfs:label ?component.
} orderby desc(?rank)
```

	task	priority	component
1	"Air System - Filter Renew"	"SAFETY CRITICAL"	"Air System"
2	"Safety Systems - Test"	"SAFETY CRITICAL"	"Safety Systems"
3	"Air System - Dryer Unit - Renew"	"MISSION CRITICAL"	"Air System"
4	"Air System - Dryer Unit Filter Renew"	"MISSION CRITICAL"	"Air System"

Figure 6: Tasks ordered by the estimated risk.

6. Analysis and Validation of Acquired Knowledge

In this section, we discuss the nature of the knowledge captured in RailMain-KG and how it may be validated. Through a brief manual validation, we assess the quality of the generated information during the knowledge acquisition pipeline. Furthermore, we describe the maintenance plans and provide some interesting insights on how the proposed resource can impact contemporary planning problems in the train maintenance context.

6.1. Size of RailMain-KG

We applied the acquisition procedure to a particular manufacturer during a recent exercise. The RailMail-KG includes information about 250 maintenance tasks that are classified across 19 unique activity types and covers 65 components. Each task is assigned to at most three vehicle types, marked with a priority rank and contains links to maintenance instructions and safety warnings. There are references to 777 resource requirements aligned to 212 unique materials and 154 tools. Overall, information within RailMain-KG is encoded in 7,610 triples.

6.2. Validation Against Manual Encodings

Entities are automatically extracted and linked to formal concepts during the knowledge acquisition pipeline. The quality of the generated content is assessed by comparing a subset of VMIs against information encoded within RailMain-KG. To this end, 20 randomly selected VMIs were manually digitised based on the thematic sections introduced in Section 4.2. This information was then compared to triples within RailMain-KG and the results are summarised in a confusion matrix (Table 1). The validation indicates that the proposed pipeline performs

Table 1
Comparison of 20 manually extracted VMIs against RailMain-KG. TP: true positive (correctly identified element), FP: false positive (incorrectly identified element), FN: false negative (missing element). F1: f1 score (correlation of precision and recall).

Feature	TP	FP	FN	Precision	Recall	F1
Task name	20	0	0	1	1	1
Component	20	0	0	1	1	1
Priority	20	0	0	1	1	1
Activity type	22	0	0	1	1	1
Vehicle	60	0	0	1	1	1
Material	45	3	8	0.94	0.85	0.89
Tool	38	3	0	0.93	1	0.96
Average				0.98	0.98	0.98

as required in extracting information about vehicles, components, priorities and so on. In the case of resource identification, on the other hand, 94% of materials and 93% of tools within the sample VMIs are correctly linked to industrial concepts; in terms of recall, 85% of the necessary materials are included in the RailMain-KG triples. The incorrect linkage between resources and industrial concepts is primarily due to strict thresholds used to parameterise elastic search coupled with spelling variations between resource mentions in VMIs and terms within industrial vocabularies. In terms of missing resources, the inconsistent structure of VMI documents impedes XSLT transformations resulting in incomplete information extraction. Despite the factors affecting performance, overall, RailMain-KG achieves an average F1 score of 0.98%, which is quite impressive given the fact that the knowledge acquisition pipeline does not benefit from the predictive capabilities of deep learning and trained language models.

6.3. Generalisation

The current version of the graph is built on VMIs provided by a specific manufacturer. Consequently, the knowledge acquisition pipeline is compatible with VMIs framed on a particular template and format. The rolling stock maintainer may change during the rolling stock life cycle (e.g., application of the manufacturer maintenance manuals during the first two years of the rolling stock), which raises the need for transferability. Since there is no industry wide standard structure for VMIs, the proposed methodology requires parsers to accommodate the structure of unseen VMI templates. In particular, information extraction and entity linking require a third-party component to divide and organise the textual content of VMIs into themed sections as they are described in Section 4.2. Entity linking mechanisms that identify resources (elastic search) can be applied on text introduced by parsers assuming it is framed in a semi-structured format. This assumption is due to the fact that to the best of the authors' knowledge, there is currently no language model that allows entity recognition from free-form industrial-themed text. We intend to lift this assumption in future research as described in Section 7. In the case of regular expressions, generic patterns, such as text segmentation, can be reapplied out of the box; however, patterns for identifying significance labels, actions and mechanical systems might require tuning to capture the intended information effectively. For instance, they are limited in identifying keywords, which are found within the data set that was used to build RailMain-KG. Finally, investing further in transferability, one of the benefits of elastic search, when used to link text to industrial concepts, is that it does not rely on annotated corpus or pretrained language models. Therefore, acquired information such as components, priority labels, vehicles, as well as aliases of known resources are utilised to introduce or extend existing non-metric indices, which in turn enhances the linking capabilities of the proposed pipeline.

7. Conclusions

Creating rich digital models of industrial processes is seen as a step towards realising the benefits in movements such as Industry 4.0. In our project, we have been engaged in creating a Virtual Depot for capturing the processes, knowledge and data involved in train vehicle maintenance, and in particular in an attempt to integrate scheduling and planning processes within the Depot. A major challenge in this pursuit is in the knowledge acquisition of detailed company-specific knowledge and data about maintenance tasks, resources, tools etc., which are required for input to these processes. In this paper, the authors have described a novel ontology-driven process which creates a digitised copy (RailMain-KG) of the information found in a sample set of VMIs. Textual information within maintenance instructions is converted into a graph-like structure with well-defined semantics, allowing information retrieval based on logical relations instead of statistically driven text similarity. Section 5 briefly demonstrates the benefits of semantic search over RailMain-KG, where text information encoded as semantically-rich concepts can feed the maintenance scheduling process and further advance automation. Our work is ongoing: while we have created a short-term scheduling operation which draws on knowledge in the RailMain-KG, we have yet to animate the end-to-end maintenance process[13].

There are factors affecting the performance of the introduced pipeline in identifying and linking terminology with industrial concepts. We plan to replace the strict pattern matching of regular expressions by annotating maintenance instructions found in RailMain-KG with the goal of training a language model specialised in named entity recognition of industrial terminology within the domain of rolling stock maintenance. Regarding linking terms with industrial concepts, a meticulous curation of the industrial vocabularies is in the proposed pipeline; it intends to dissolve spelling discrepancies and also express hierarchical relations between concepts. Other work will focus on condition-based maintenance scheduling processes where the thresholds related to some maintenance intervention such as wheel flat length or depth will be automatically extracted from VMIs to build an automated decision tool for the maintenance scheduling process. To move from periodic preventive maintenance to condition-based one, the thresholds, controlling when certain activities should be done, are required. Some of these thresholds are standardised in the context of rolling stock maintenance such as wheelset-related thresholds but many others are not. Each maintainer has its own threshold values that are specified in VMIs.

Acknowledgments

This work is part of the Smart Rolling Stock Maintenance Research Facility project, funded by the European Regional Development Fund and Northern Powerhouse.

References

[1] H. Louadah, E. Papadakis, L. McCluskey, G. Tucker, P. Hughes, A. Bevan, Translating ontological knowledge to pddl to do planning in train depot management operations, in: 36th Workshop of the UK Planning and Scheduling Special Interest Group, 2021.

[2] Y. Gil, Description logics and planning, AI magazine 26 (2005) 73–73.

[3] E. Bouillet, M. Feblowitz, Z. Liu, A. Ranganathan, A. Riabov, A knowledge engineering and planning framework based on owl ontologies, 2007.

[4] M. Cioffi, S. G. Thompson, Planning with the semantic web by fusing ontologies and planning domain definitions, in: Research and Development in Intelligent Systems XXIII, Proceedings of AI, Springer, 2006.

[5] A. V. Feljan, A. Karapantelakis, L. Mokrushin, H. Liang, R. Inam, E. Fersman, C. R. B. Azevedo, K. Raizer, R. S. Souza, A framework for knowledge management and automated reasoning applied on intelligent transport systems, CoRR (2017).

[6] P. Bonasso, M. Boddy, D. Kortenkamp, S. Bell, Ontological models to support planning operations, in: Proceedings of 8 th International Workshop on Planning and Scheduling for Space, 2013.

[7] I. Gocev, S. Grimm, T. A. Runkler, Explanation of action plans through ontologies, in: On the Move to Meaningful Internet Systems. OTM 2018 Conferences, Springer International Publishing, 2018.

[8] C. Li, X. Yang, S. Luo, M. Song, W. Li, Towards domain-specific knowledge graph construction for flight control aided maintenance, Applied Sciences (2022).

[9] L. Xia, Y. Liang, J. Leng, P. Zheng, Maintenance planning recommendation of complex industrial equipment based on knowledge graph and graph neural network, Reliability Engineering & System Safety (2023).

[10] X. Liu, H. Wang, Knowledge graph construction and decision support towards transformer fault maintenance, in: 2021 IEEE 24th International Conference on Computer Supported Cooperative Work in Design (CSCWD), IEEE, 2021.

[11] Z. Ragala, A. Retbi, S. Bennani, Construction and application of knowledge graph for railway rolling stock failure tree, in: 2022 IEEE International Conference on Technology Management, Operations and Decisions (ICTMOD), IEEE, 2022.

[12] L. Boytsov, B. Naidan, Engineering efficient and effective non-metric space library, in: N. R. Brisaboa, O. Pedreira, P. Zezula (Eds.), Similarity Search and Applications - 6th International Conference, SISAP 2013, Proceedings, volume 8199 of *LNCS*, Springer, 2013, pp. 280–293. doi:10.1007/978-3-642-41062-8_28.

[13] H. Louadah, T. L. McCluskey, G. Tucker, I. Hamaz, E. Papadakis, Towards rolling stock preventive maintenance scheduling - short-term scheduling optimisation, Scheduling and Planning Applications woRKshop (SPARK), 2023.

[14] M. F. M. Idris, N. H. Saad, M. I. Yahaya, A. Shuib, W. M. W. Mohamed, A. N. M. Amin, Cost of rolling stock maintenance in urban railway operation: Literature review and direction, Pertanika Journal of Science & Technology (2022).