World Academy of Science, Engineering and Technology
International Journal of Industrial and Manufacturing Engineering
Vol:9, No:1, 2015

A Temporal QoS Ontology for ERTMS/ETCS

Marc Sango, Olimpia Hoinaru, Christophe Gransart, Laurence Duchien

***Abstract*—**Ontologies offer a means for representing and sharing information in many domains, particularly in complex domains. For example, it can be used for representing and sharing information of System Requirement Specification (SRS) of complex systems like the SRS of ERTMS/ETCS written in natural language. Since this system is a real-time and critical system, generic ontologies, such as OWL and generic ERTMS ontologies provide minimal support for modeling temporal information omnipresent in these SRS documents. To support the modeling of temporal information, one of the challenges is to enable representation of dynamic features evolving in time within a generic ontology with a minimal redesign of it. The separation of temporal information from other information can help to predict system runtime operation and to properly design and implement them. In addition, it is helpful to provide a reasoning and querying techniques to reason and query temporal information represented in the ontology in order to detect potential temporal inconsistencies. To address this challenge, we propose a lightweight 3-layer temporal Quality of Service (QoS) ontology for representing, reasoning and querying over temporal and non-temporal information in a complex domain ontology. Representing QoS entities in separated layers can clarify the distinction between the non QoS entities and the QoS entities in an ontology. The upper generic layer of the proposed ontology provides an intuitive knowledge of domain components, specially ERTMS/ETCS components. The separation of the intermediate QoS layer from the lower QoS layer allows us to focus on specific QoS Characteristics, such as temporal or integrity characteristics. In this paper, we focus on temporal information that can be used to predict system runtime operation. To evaluate our approach, an example of the proposed domain ontology for handover operation, as well as a reasoning rule over temporal relations in this domain-specific ontology, are presented.

***Keywords*—**System Requirement Specification, ERTMS/ETCS, Temporal Ontologies, Domain Ontologies.

I. Introduction

THE System Requirement Specification (SRS) documents are generally written in natural language. From these documents, ontologies offer a means for representing, sharing and querying information in many complex domains, particularly in critical systems as proposed in automotive [1], avionic [2] and biomedical domains [3]. Similar works, such as [4], [5] and [6] are proposed in the railway domains with the aim of formalizing the SRS documents of the new European Rail Traffic Management System ERTMS including automatic train control and signaling system ETCS [7]. However, these generic ontologies provide minimal support for modeling and querying temporal information omnipresent in these SRS documents.

Marc Sango is Ph.D. candidate in Computer Science from the University of Lille 1 and IFSTTAR, France. (e-mail: marc.sango@ifsttar.fr).
Olimpia Hoinaru is engineer at IFSTTAR, France. (e-mail:olimpia.hoinaru@ifsttar .fr).
Christophe Gransart is senior researcher at IFSTTAR, France. (e-mail: christophe.gransart@ifsttar.fr).
Laurence Duchien is Professor of Computer Science at the University of Lille 1, France. (e-mail: Laurence.Duchien@univ-lille1.fr).

To support the modeling of temporal information, one of the challenges is to enable representation of dynamic features evolving in time within a generic ontology with a minimal redesign of it. The separation of temporal information from other information can help to predict system runtime operation and to properly design and implement them [8]. In addition, it is helpful to provide a reasoning and querying technique for inferring temporal information represented in the ontology in order to detect potential temporal inconsistencies [9]. Indeed, a user operation, such as adding a new constraint on existing planning constraints, can cause temporal inconsistencies, which can lead to system failures.

To address this challenge, we propose a lightweight 3-layer temporal Quality of Service (QoS) ontology for representing, reasoning and querying over temporal and non-temporal information in a complex domain ontology. Representing QoS entities in separated layers can clarify the distinction between the non QoS entities and the QoS entities in an ontology. The upper generic layer of the proposed ontology provides an intuitive knowledge of domain components. Our temporal QoS ontology has been designed to be integrated in an existing ERTMS ontology [6] with a minimal redesign of it. The separation of intermediate QoS layer from the lower QoS layer allows us to focus on specific QoS Characteristics, temporal or integrity. In this paper we focus on temporal information that help us to predict system runtime operation. To evaluate our approach, the proposed domain ontology is applied for handover operation, i.e., the handing over of train control between two Radio Block Centers (RBCs) at the RBC-RBC borders. Then, using reasoning rules, we infer implicit knowledge of temporal information in this domain-specific ontology.

The remainder of this paper is organized as follows. In Section II, the ontology background is introduced. In Section III, the temporal QoS ontology developed to represent and query temporal and non-temporal information in ERTMS/ETCS is presented. Then, in Section IV, the proposed domain ontology is instantiated and evaluated for the domain-specific RBC-RBC handover procedure. Finally, we discuss our approach in Section V before concluding and drawing future directions of our research in Section VI.

II. Background on Ontology

Ontologies are widely used to represent knowledge, by describing data in a formal and explicit manner and then play a key role in the evolution of the semantic particularly the Semantic Web initiative [10]. The Ontology Web Language (OWL) [11] is the W3C recommendation for creating and sharing ontologies in the Web. Its theoretical background is based on the Description Logic (DL) knowledge

representation formalism [12]. OWL is a richer vocabulary description language for representing properties and classes, such as relations between classes (e.g. disjointness), cardinality (e.g. exactly one), equality, richer typing of properties, characteristics of properties (e.g. symmetry) and enumerated classes [13]. In addition, the formal semantics of the OWL language enable the application of relation reasoning techniques performed by rule engines like the Jess Rule Engine [14] or by reasoners such as KAON2 [15] or Pellet [16]. Then, tools are introduced to solve problems related to knowledge representation and reasoning. For example, Protégé[1] is an open source ontology editor which allows the creation of OWL DL based ontologies and supports reasoners Pellet to interpret the rules like the Semantic Web Rule Language (SWRL) [17]. Furthermore, It also allows the ability for extracting and querying information from the asserted data in ontology. For example, SWRL-based query language called SQWRL provides a simple yet expressive language for performing queries on OWL ontologies [18].

The OWL-Time temporal ontology [19] describes the temporal content of OWL-based web Service (OWL-S [20]) and the temporal properties based on Allen's interval-based relations [21]. In this ontology, there are two subclasses of temporal entities: interval and instant, where an instant is an interval with zero length. However, OWL-Time is not an ontology language, but a time ontology based on OWL [22]. There is a large amount of research for temporal ontology languages based on OWL [23], [24], [22], [25]. However, representing temporal information requires support for concrete domains [26]. The concrete domain approach [27] requires introducing additional datatypes and operators to OWL, while our work relies on existing ERTMS/ETCS ontology with minimal redesign of it. Our approach is comparable to a novel approach for connecting temporal-ontologies with blood flow simulations [28]. In this approach, an existing Basic Formal Ontology (BFO [29]) and Relations in Biomedical Ontologies(RBO [30]) are consolidated with the SWRL Temporal Ontology (SWRLTO [31]) to query temporal information in the Temporal BFO result. While the technique is the same, the problem domain we explore is really different. In our approach, we use the generic ERTMS ontology [6] and the temporal QoS Ontology including the W3C N-ary relations [32] to develop a temporal QoS ontology for ERTMS/ETCS.

III. Temporal QoS Ontology for ERTMS/ETCS

In this section, the temporal QoS ontology developed to represent and query temporal and non-temporal information in ERTMS/ETCS is introduced.

A. ERTMS/ETCS Generic Ontology

Our temporal QoS ontology is integrated on existing generic non-temporal ontology for ERTMS [33], [6]. The aim of this generic ontology is the formalization of ERTMS/ETCS SRS documents [7] in order to obtain a component structure that can be reusable in a development process. Since the first focus is a formalization feasibility, this generic ontology presents only a representation view of functional requirements of the SRS. The functional requirements are related to (i) *trackside* and *onboard* train sub-systems of ETCS including, among others RBC[2] and odometry, (ii) their *interfaces* including, among others, train, driver and interlocking[3], (iii) the five specified *application levels* to express the possible operating relationships between trackside and onboard train sub-systems, (iv) *procedures* including, among others, start of mission or RBC-RBC handover procedure described with state transition system and (v) communication *language* which includes radio or balise messages built by packet, telegram and variables.

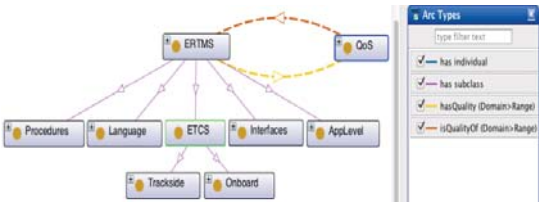

Fig. 1 ERTMS/ETCS Generic Ontology

Fig. 1 shows the hierarchy of top-level class categories of this generic ontology redesigned. The super-class *ERTMS* represents the European Rail Traffic Management System, while *ETCS* for European Train Control System represent the signaling subsystem composed by the *Onboard* sub-class and the *Trackside* sub-class. It also includes the *Procedures* sub-class, the *Interfaces* sub-class, the application level (*AppLevel*) sub-class and the *Language* sub-class.

This redesigned generic ERTMS ontology currently contains 112 classes and subclasses, 193 instances, and 104 properties including objects, datatypes and annotation properties [6]. Some of these properties are shown in the right-hand part of Fig. 1. Compared to the available SRS documents, the current coverage of the available SRS by the ERTMS ontology entities is obviously reduced. Nonetheless, it represents the general high-level functionalities that can be specialized in order to make ontological results really usable. The shortcoming of this generic ontology is the lack of non-functional Quality of Service (QoS) characterization, such as temporal QoS that specifies temporal quality criteria that can be used as a prediction of system runtime operation. As a consequence, we extend the *ERTMS* class with the QoS Class as illustrated in Fig. 1. In addition to *has subclass* and *has individual* properties that represent the relation among classes and individuals (i.e., instances of class), *hasQuality* and its inverse *isQualityOf* are added to represent the QoS properties. In the following, we detail the integration of this QoS Ontology.

[1] http://protege.stanford.edu

[2] Radio Block Centre

[3] Underlying signaling system for train integrity supervision

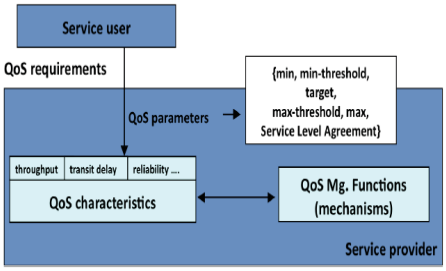

Fig. 2 ITU X.641 QoS Framework

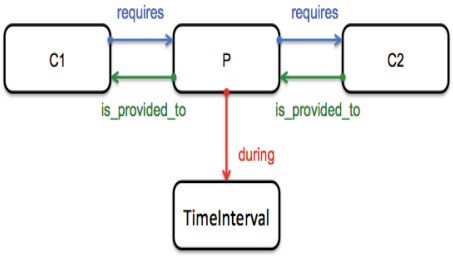

Fig. 3 N-ary Relations Example

B. QoS Ontology for ERTMS/ETCS

The QoS ontology developed for the ERTMS ontology is based on the generic QoS Framework as recommend by the ITU-T Recommendation X.641 [34] and its derivation for the next transport generation QoS ontology [35]. In this recommendation, the term "service" in the expression "Quality of Service" is to be understood as a very general term, such as processing, storing, transmitting and delivering, in order to permit the widest possible application of the QoS Framework. Fig. 2 illustrates the main concepts of this QoS Framework. These concepts include *service user, service provider, QoS requirements, QoS characteristics, QoS parameters, QoS management functions*.

A *service provider* is an entity (e.g., a component with output ports) responsible to deliver a service to the *service user* (e.g., a component with input ports). *QoS requirements* are a part of user requirement specification, i.e., conditions or constraints under which the solution of functional requirement must operate. It is defined as high-level constraints in order to manage one or several QoS characteristics. *QoS characteristics* are defined as quantifiable aspects of service provider, such as temporal transit delay or reliability aspect. A QoS characteristic is described by QoS parameters. *QoS parameters* are defined as vectors of scalar values, such as the range between a maximum or minimum level of a transit delay characteristic. QoS management functions, *QoSMgFunctions*, refer to the activities for maintaining the quality parameters in the required range if the quality change. For example, if a quality of a component changes then a new instance of the management functions in which the component participates is initialized with the new quality values. In this way, there are the fundamental temporal relations which could hold among instantaneous objects (i.e., component instances) during a specific time instant or among interval of function instances themselves. These temporal relations can be analyzed with the lower-level techniques, such as a real-time scheduling theory [36]. In addition to lower level techniques, in this paper, we claim that the temporal knowledge can be directly handled in domain ontologies in order to facilitate much higher-level analysis and to make better sense of complex temporal patterns for software developers. As a consequence, in the following, we will focus on the temporal characteristics of the QoS ERTMS/ETCS ontology.

C. Temporal QoS Ontology for ERTMS/ETCS

For the purpose of adding a valid temporal QoS dimension to above-mentioned ontology, we use the N-ary relations approach [32] in addition to the classical OWL ontology binary relations between two elements (e.g., *has subclass*). In fact, temporal relations are properties of objects that change in time and involve also a temporal value in addition to the objects. They are ternary relations that cannot be expressed directly in binary relations. The N-ary relations approach suggests representing an N-ary relation as two properties each related to a new object, which in turn is related with the temporal interval that this relation holds. For example, Fig. 3 illustrates the representation of two temporal properties using the N-ary approach: the *requires* property which states that "Component C1 requires Component C2 during TimeInterval" and the *is_provided_to* property which states that "Component C2 is provided to Component C1 during TimeInterval". These two properties are related to a process P object, which in turn is related to the *TimeInterval* through a temporal relation *during*.

Fig. 4 shows our 3-layer temporal QoS ontology for ERTMS. The upper layer shows the generic ERTMS ontology that represents the reusable components of ERTMS. The intermediate layer describes the generic QoS characteristics of these components. The lower layer represents the specific temporal QoS characteristics layer. Representing temporal entities in separated layers can clarify the distinction between the non temporal entities and the temporal entities in an ontology. In the generic ontology, the most significant component that can extend over time is the *Procedures* class. In the ERTMS/ETCS SRS documents, it is described by flowchart [7], which is represented in our ontology by states, conditions and transitions. This class can be specialized in subclasses such as the handover or start of mission procedure flowchart.

Following the N-ary relations approach, a sub-class *ValidTime* is added in temporal QoS layer and the property *hasValidTime* links the *Procedures* class to *ValidTime* class, as shown in the ontology Fig. 4. The *ValidTime* sub-class is a subclass of QoS *Parameters*, which, as mentioned in Section III-B, is a vector of scalar values for QoS *Characteristics*. The *Interval* and *Instant* classes are introduced as subclasses of a superclass *Temporal* QoS characteristics. By using Allen's temporal interval relations [21], the temporal representation was enhanced with qualitative temporal relationships between two time intervals characterized by the starting and ending

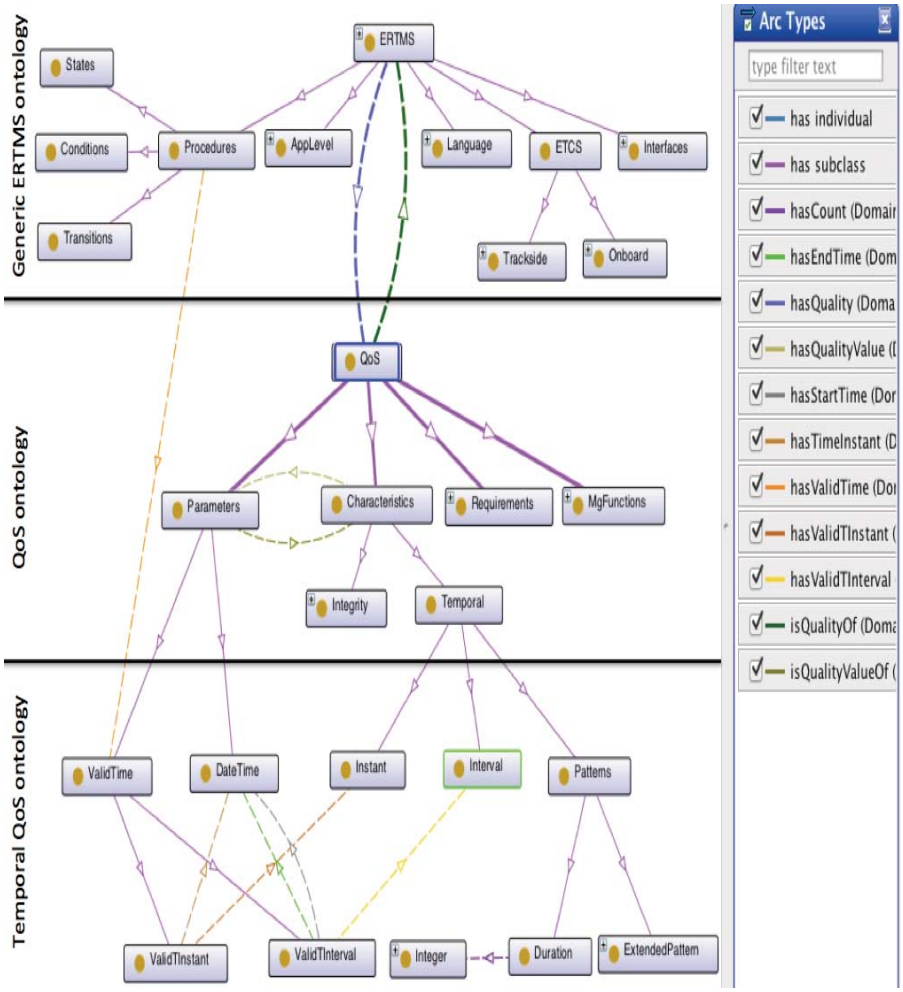

Fig. 4 3-Layer Temporal QoS Ontology for ERTMS/ETCS

points. These are represented in the ontology by the properties *hasTimeInstant*, *hasStartTime* and *hasEndTime*. Table I shows the overview of 13 Allen's relations, where $t_i = (t_i^s, t_i^e)$ and $t_j = (t_j^s, t_j^e)$ are two intervals. The intervals are characterized by the starting points t_i^s, t_j^s and the end points t_i^e, t_j^e. The end point relations $t_i^s < t_i^e$ and $t_j^s < t_j^e$ are provided for all 13 Allen relations. The relations *after*, *overlappedby*, *metby*, *contains*, *startedby* and *finishedby* are the inverse of *before*, *overlaps*, *meets*, *during*, *starts* and *finishes*. The relation *equals* is symmetric and relations *before* and *after* are transitive.

Once all temporal information is consistently represented in the ontology, it can then be manipulated by using temporal operators, such as the above-mentioned Allen's operators. The SWRL rule language [17] provides a means for defining these operators as rules to manipulate time values of temporal type, such as the XML Schema temporal types (e.g., *xsd:dateTime*, its value example is *2014-11-06T17:35:00*). We have used SWRL to implement Allen interval-based temporal operators over the *DateTime* class temporal types, as illustrated in Fig. 4. Our implementation can be extended to support other temporal types. As a consequence, we add a *Patterns* class which constrains the value space of a specific temporal type such as a duration of time (e.g. *2 Seconds*). Temporal pattern duration are represented by the class *Duration*, which has an integer functional property *hasCount* and a functional *hasTimeUnits* property which has instances, such as *Years, Months, Days, Hours, Minutes, Seconds, and Milliseconds*.

TABLE I
OVERVIEW TO ALLEN'S RELATIONS BETWEEN PAIRS OF TWO INTERVALS

Type	Meaning	Relation
$before(t_i, t_j)$	t_i before t_j	$t_i^e < t_j^s$
$after(t_j, t_i)$	t_j after t_i	
$meets(t_i, t_j)$	t_i meets t_j	$t_i^e = t_j^s$
$metBy(t_j, t_i)$	t_j met by t_i	
$overlaps(t_i, t_j)$	t_i overlaps t_j	$(t_i^s < t_j^s < t_i^e) \wedge (t_i^e < t_j^s)$
$overlapsBy(t_j, t_i)$	t_j overlaps by t_i	
$during(t_i, t_j)$	t_i during t_j	$(t_i^s > t_j^s) \wedge (t_i^e < t_j^e)$
$includes(t_j, t_i)$	t_j includes t_i	
$starts(t_i, t_j)$	t_i starts t_j	$(t_i^s = t_j^s) \wedge (t_i^e < t_j^e)$
$startedBy(t_j, t_i)$	t_j started by t_i	
$finishes(t_i, t_j)$	t_i finishes t_j	$(t_i^s > t_j^s) \wedge (t_i^e = t_j^e)$
$finishedBy(t_j, t_i)$	t_j finished by t_i	
$equals(t_i, t_j)$	t_i equals t_j	$(t_i^s = t_j^s) \wedge (t_i^e = t_j^e)$

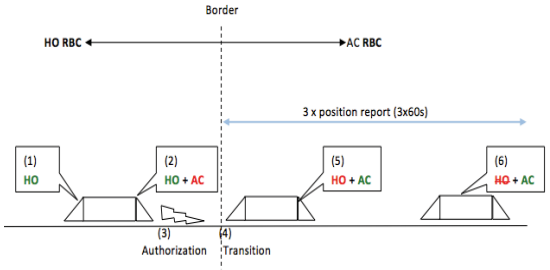

Fig. 5 Example of RBC-RBC Handover Scenario

IV. Evaluation Example

In this section, a handover procedure example, as well as an example for querying temporal information in the ontology, are presented and evaluated.

A. RBC-RBC Handover Procedure Example

When the train is moving away from the communication area covered by one Radio Block Centre (RBC) (called Handing Over RBC or HO RBC) and entering the area covered by another RBC (Accepting RBC or AC RBC), the handover procedure is called in order to transfer the communication to an AC RBC and to avoid communication termination when the train gets outside the range of the HO RBC. In the nominal operation, see Fig. 5, the main functional steps needed for running from one RBC area to another one are the following: (1) pre-announcement of the transition by the HO RBC; (2) establishment of the radio communication session with the AC RBC; (3) generation of movement authorities including the border; (4) announcement of the RBC transition (5) transfer of train supervision to the AC RBC; (6) Termination of the session with HO RBC. More detail of procedure natural language description can be found in paragraph §5.15 of [7].

Here, we will focus on one degraded situation where the time interval for running the handover can be changed. For example "in case a communication session is established and no request to terminate the session is received from the HO RBC within a fixed waiting time after sending the position report, the position report shall be repeated with the fixed waiting time after each repetition" (§5.15.4 [7]). For our evaluation, we allow 3 possible repetitions of position report and each position report takes *60 s*, as illustrated in Fig. 5. Adding this new temporal constraint, a temporal inconsistency could arise if the associated constraint conflicts with the nominal time interval constraint. In such cases, it is helpful to query the temporal information in the formalized ontology to explain if the operation is consistent or not.

B. Handover Procedure Ontology

To model this domain specific knowledge the most important first step is to instantiate the *Procedures* class with the individual objects related to the scenario. More precisely, the class *Procedures* of the first layer of Fig. 4 is extended with two instances *handhover_Process1* related to the nominal mode and *handhover_Process2* related to the degraded mode. The second step is to add concrete QoS parameter values to the instance. For this, the *hasQualityValue* property is used to enrich the QoS ontology characteristic with the ability to add concrete parameter values. Here, the request for position report of the *handhover_Process2* degraded situation is set to 3x60s as illustrated in Fig. 5.

The last step is to represent temporal information by using the temporal class *ValidTime* and subclass *ValidTInstant* and *ValidTInterval* with the properties associated. For example, the nominal handover starting date time is set to *2014-11-06T17:35:00* and the ending dated to *2014-11-06T17:40:00*. As explained so far, this date information is stored as a literal of the XML schema type *xsd:dateTime*. This type of information cannot be included in the deductive reasoning process with reasoners, such as previously mentioned Pellet reasoner integrated in Protégé. One way to query this temporal information is to use the SWRL-based Query language (SQWRL) [18].

TABLE II
A SWRL Rule

$$
\begin{aligned}
& ertms : procedure(?p) \\
\wedge\ & temporal : hasValidTime(?p, ?pVT) \\
\wedge\ & temporal : hasStartTime(?pVT, ?pST) \\
\wedge\ & temporal : hasFinishTime(?pVT, ?pET) \\
\rightarrow\ & sqwrl : select(?p, ?pST, ?pET)
\end{aligned}
$$

TABLE III
A SQWRL Query Result

individuals (id)	start time (st)	end time (et)
ertms:handhover_Process1	2014-11-06T17:35:00	2014-11-06T17:40:00
ertms:handhover_Process2	2014-11-06T18:35:00	2014-11-06T18:43:00

C. Query of Temporal Information

We use Allen basic interval operators, implemented with the SWRL rules, and the SQWRL to query the temporal information in the ontology. In fact, SQWRL is defined as a set of SQL-Like query operators, such as *sqwrl:select*. Concretely this SQL-Like query operator is used as a consequent of a SWRL rule to format the information and makes it possible to infer implicit knowledge from the temporal information. For example, Table II shows a query, where variable $?p$ represents the process instances represented in the handover ontology and variables $?pST$ and $?pET$ represent their start and end time respectively.

The result of the query is shown in Table III. It shows the start and the end time of the nominal handover *handhover_Process1* and the degraded *handhover_Process2*. Actually, only Allen basic operators are used in the query but we are looking for more advanced temporal queries, as proposed in [31].

V. Discussion

Temporal information can be analyzed with the lower-level techniques, such as a real-time scheduling theory [36]. In addition to lower-level techniques, in this paper, we claim that the temporal knowledge can be directly handled in domain

ontologies in order to facilitate much higher-level analysis and to make better sense of complex temporal patterns for software developers. In fact, working at the knowledge level will enable software developers to make better sense of complex temporal patterns. The aim of our approach is to enhance the ability of software developers to clearly understand complex forms of temporal knowledge in order to properly implement them. Similar domain ontologies have been proposed in other critical domains such as automotive [1], avionic [2] and clinical domains [28]. In contrast to previous works on ERTMS/ETCS [4], [5], [6], which do not focus on temporal constraints or which required some rewriting of source ontology to integrated temporal information, our approach provides separation of concerns with a 3-layer temporal QoS ontology.

Regarding the representation and reasoning abilities of temporal information in temporal ontologies and based on some general requirements of the handbook of temporal reasoning [37], the following requirements are identified.

1) *Instants/events*: *Instants* are represented by the class *temporal:Instant*. Events can be modeled under the class *Procedures*. Events are characterized by instants stating the time the event occurred.
2) *Intervals/processes*: *Intervals* are represented by the class *Temporal:Interval* with the property *hasValidTInterval*. A process, represented by the class *Procedures*, is characterized with intervals by stating start and end time of the process.
3) *Scheduling*: To represent *scheduling processes*, the process itself and the scheduling algorithm which regulates the periodicity have to be modeled. The former can be modeled in our ontology by specializing the class *Procedures* but the latter cannot be represented in the actual version of our ontology.
4) *Causality*: With the Allen relations representation, our ontology comes with a small set of temporal relations which enable basic representation of causality in terms of interval relations. In addition, more advanced temporal relations can be used by composing the basic relations.
5) *Patterns*: Defining temporal patterns is a central requirement when working with temporal aspects. A pattern is defined as compounded processes consisting of simpler processes in order to provide more precision. In our ontology it is possible to represent them by using the class *Patterns*.
6) *Transformation to formal model*: The connection of our ontology to a formal model is not yet provided in the current version. But we are looking for its possible connection to a formal time automata model for which we can use the appropriate formal model checker tool, such as UPPAAL for formal verification tasks of temporal safety properties.
7) *Traceability*: The traceability must be established between source requirement documents and ontology entities. In our ontology, this is simply provided by associating to the ontology entities the requirement text as a comment. However, if the source ontology must be transformed to the valid time formal model, a rigorous traceability is required.
8) *Coverage* Compared to the complexity of the ERTMS/ETCS SRS, the current coverage of the available SRS by the generic ontology layer is reduced. But it represents the general high-level functionalities that can be specialized in order to make ontological results really usable. The temporal QoS ontology layer covers the most common temporal information met in these SRS documents. It can be specialized for the complete coverage of a domain-specific area, such as the rail-road level crossing area subjected to strong temporal constraints.

VI. Conclusion and Future Work

The main challenge we face in this paper is how to integrate the temporal QoS dimension in existing ontologies with a minimal redesign of them. In order to face this challenge, we introduce a lightweight 3-layer temporal QoS ontology developed for ERTMS/ETCS. This development is mainly based on the study of a set of referential SRS documents of ERTMS/ETCS. Representing QoS entities in separated layers can clarify the distinction between the non QoS entities and the QoS entities in an ontology. The upper generic layer of the proposed ontology provides an intuitive knowledge of ERTMS/ETCS domain components. The separation of the intermediate QoS layer from the lower QoS layer allows us to focus on specific QoS Characteristics, such as temporal of integrity characteristics. In this paper we focus on temporal information. For evaluation, we instantiate the proposed ontology for RBC handover procedure and use a rule-based query language SQWRL to infer temporal information in this domain-specific ontology.

A potential shortcoming of our approach is that temporal information in the ontology may be transformed into the formal model in order to verify temporal safety properties for high-integrity levels. This model-driven transformation process can be error-prone. As a consequence, our future work will focus on an ontology bridge in model-driven transformation for safety assessment. Clearly, we are looking to create a bridge between the proposed ontology and a time automata formal model, such as [38], for which we can use the appropriate model checker tool, such as UPPAAL for formal verification tasks of temporal safety properties. In addition, this process can require considerable safety domain-specific expertise. As a consequence, and additional future work is to demonstrate the efficiency of our approach with railway specific scenarios, such as an operational scenario proposed for the new safer rail-road level crossing architecture subjected to strong temporal safety properties [39].

References

[1] M. Feld and C. Müller, "The automotive ontology: Managing knowledge inside the vehicle and sharing it between cars," in *Proceedings of the 3rd International Conference on Automotive User Interfaces and Interactive Vehicular Applications*, ser. AutomotiveUI '11, 2011, pp. 79–86.

[2] S. Farfeleder, T. Moser, A. Krall, T. Stlhane, H. Zojer, and C. Panis, "Dodt: Increasing requirements formalism using domain ontologies for improved embedded systems development," in *Design and Diagnostics of Electronic Circuits Systems*, April 2011, pp. 271–274.

[3] OBO, "The open biological and biomedical ontologies," http://www.obofoundry.org/.
[4] S. Verstichel, F. Ongenae, L. Loeve, F. Vermeulen, P. Dings, B. Dhoedt, T. Dhaene, and F. D. Turck, "Efficient data integration in the railway domain through an ontology-based methodology," *Transport. Research Part C: Emerging Technologies*, vol. 19, no. 4, pp. 617 – 643, 2011.
[5] G. Bonifacio, P. Marmo, A. Orazzo, I. Petrone, L. Velardi, and A. Venticinque, "Improvement of processes and methods in testing activities for safety-critical embedded systems," in *Computer Safety, Reliability, and Security*, 2011, vol. 6894, pp. 369–382.
[6] O. Hoinaru, C. Gransart, G. Mariano, and E. Lemaire, "An ontology for the ERTMS/ETCS," in *Transport Research Arena*, Paris, 2014, p. 10p.
[7] ERTMS/ETCS, http://www.era.europa.eu/Document-Register/Pages/SystemRequirementsSpecification(Recommendation).aspx.
[8] M. Sango, C. Gransart, and L. Duchien, "Safety component-based approach and its application to ERTMS/ETCS on-board train control system," in *Transport Research Arena*, Paris, Apr. 2014, p. 10.
[9] J. L. Bresina and P. H. Morris, "Explanations and recommendations for temporal inconsistencies," *Proc. Int. Work. on Planning and Scheduling for Space*, 2006.
[10] D. L. McGuinness and F. van Harmelen, "The semantic web activity."
[11] W3C, "Owl web ontology language overview - w3c recommendation."
[12] F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, Eds., *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.
[13] G. Antoniou and F. v. Harmelen, *A Semantic Web Primer, 2Nd Edition (Cooperative Information Systems)*, 2nd ed. The MIT Press, 2008.
[14] JESS, http://www.jessrules.com/.
[15] KAON2, http://kaon2.semanticweb.org.
[16] E. Sirin, B. Parsia, B. C. Grau, A. Kalyanpur, and Y. Katz, "Pellet: A practical owl-dl reasoner," *Web Semant.*, vol. 5, no. 2, pp. 51–53, Jun. 2007.
[17] SWRL, "Swrl: A semantic web rule language combining owl and ruleml - w3c member submission 21 may 2004."
[18] M. J. O'Connor and A. K. Das, "Sqwrl: A query language for owl." in *OWLED*, ser. CEUR Workshop Proceedings, R. Hoekstra and P. F. Patel-Schneider, Eds., vol. 529, 2008.
[19] W3C, "Time ontology in owl - w3c working draft 27 september 2006."
[20] OWL-S, "Daml services," http://www.daml.org/services/owl-s/.
[21] J. F. Allen, "Maintaining knowledge about temporal intervals," *Commun. ACM*, vol. 26, no. 11, pp. 832–843, Nov. 1983.
[22] S.-K. Kim, M.-Y. Song, C. Kim, S.-J. Yea, H. C. Jang, and K.-C. Lee, "Temporal ontology language for representing and reasoning interval-based temporal knowledge," in *Proceedings of the 3rd Asian Semantic Web Conference on The Semantic Web*, 2008, pp. 31–45.
[23] C. Welty and R. Fikes, "A reusable ontology for fluents in owl," in *Proceedings of the 2006 Conference on Formal Ontology in Information Systems*. IOS Press, 2006, pp. 226–236.
[24] H.-U. Krieger, "Where temporal description logics fail: Representing temporally-changing relationships," in *KI 2008: Advances in Artificial Intelligence*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2008, vol. 5243, pp. 249–257.
[25] V. Milea, F. Frasincar, and U. Kaymak, "towl: A temporal web ontology language," *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 42, no. 1, pp. 268–281, 2012.
[26] S. Batsakis, K. Stravoskoufos, and E. G. M. Petrakis, "Temporal reasoning for supporting temporal queries in OWL 2.0," in *Knowledge-Based and Intelligent Information and Engineering Systems*, 2011, pp. 558–567.
[27] C. Lutz, "Description logics with concrete domains-a survey," in *Fourth conference onAdvances in Modal logic*, 2002, pp. 265–296.
[28] F. Weichert, C. Mertens, L. Walczak, G. Kern-Isberner, and M. Wagner, "A novel approach for connecting temporal-ontologies with blood flow simulations." *Journal of Biomedical Informatics*, vol. 46, no. 3, pp. 470–479, 2013.
[29] P. Grenon, B. Smith, and L. Goldberg, "Biodynamic ontology: Applying bfo in the biomedical domain," in *Stud. Health Technol. Inform.* IOS Press, 2004, pp. 20–38.
[30] B. Smith, W. Ceusters, B. Klagges, J. Khler, A. Kumar, J. Lomax, C. Mungall, F. Neuhaus, A. Rector, and C. Rosse, "Relations in biomedical ontologies," *Genome Biology*, vol. 6, no. 5, 2005.
[31] M. OConnor and A. Das, "A method for representing and querying temporal information in owl," in *Biomedical Engineering Systems and Technologies*, ser. Communications in Computer and Information Science, A. Fred, J. Filipe, and H. Gamboa, Eds. Springer Berlin Heidelberg, 2011, vol. 127, pp. 97–110.
[32] W3C, "Defining n-ary relations on the semantic web - w3c working group note 12 april 2006."
[33] O. Hoinaru, G. Mariano, and C. Gransart, "Ontology for complex railway systems application to ERTMS/ETCS system," in *FM-RAIL-BOK Workshop in SEFM2013 11th International Conference on Software Engineering and Formal Methods*, Espagne, Jan. 2013, p. 6p.
[34] ITU-T, *X.641: Information Technology - Quality of Service Frameworks*.
[35] E. Exposito, "Methodology, models and paradigms for a next generation transport layer design," Ph.D. dissertation, INPT - France, 2010.
[36] F. Singhoff, A. Plantec, P. Dissaux, and J. Legrand, "Investigating the usability of real-time scheduling theory with the cheddar project," *Real-Time Syst.*, vol. 43, no. 3, pp. 259–295, Nov. 2009.
[37] M. Fisher, D. Gabbay, and L. Vila, *Handbook of Temporal Reasoning in Artificial Intelligence (Foundations of Artificial Intelligence (Elsevier))*. New York, NY, USA: Elsevier Science Inc., 2005.
[38] M. Sango, L. Duchien, and C. Gransart, "Component-Based Modeling and Observer-Based Verification for Railway Safety-Critical Applications," in *The 11th International Symp. on FACS*, Italie, Sep. 2014, p. 18.
[39] L. Khoudour, M. Ghazel, F. Boukour, M. Heddebaut, and E.-M. El-Koursi, "Towards safer level crossings: existing recommendations, new applicable technologies and a proposed simulation model," *European Transport Research Review*, vol. 1, no. 1, pp. 35–45, 2009.