BeAware!—Situation Awareness, the Ontology-Driven Way

Norbert Baumgartner[a,*], Wolfgang Gottesheim[b], Stefan Mitsch[b], Werner Retschitzegger[c], Wieland Schwinger[b]

[a]*team Communication Technology Mgmt. GmbH, Goethegasse 3, 1010 Vienna, Austria*
[b]*Johannes Kepler University Linz, Altenbergerstr. 69, 4040 Linz, Austria*
[c]*University of Vienna, Dr.-Karl-Lueger-Ring 1, 1010 Vienna, Austria*

Abstract

Information overload is a severe problem for human operators of large-scale control systems as, for example, encountered in the domain of road traffic management. Operators of such systems are at risk to lack situation awareness, because existing systems focus on the mere presentation of the available information on graphical user interfaces—thus endangering the timely and correct identification, resolution, and prevention of critical situations. In recent years, ontology-based approaches to situation awareness featuring a semantically richer knowledge model have emerged. However, current approaches are either highly domain-specific or have, in case they are domain-independent, shortcomings regarding their reusability.

In this paper, we present our experience gained from the development of BeAware!, a framework for ontology-driven information systems aiming at increasing an operator's situation awareness. In contrast to existing domain-independent approaches, BeAware!'s ontology introduces the concept of spatio-temporal primitive relations between observed real-world objects thereby improving the reusability of the framework. To show its applicability, a prototype of BeAware! has been implemented in the domain of road traffic management. An overview of this prototype and lessons learned for the development of ontology-driven information systems complete our contribution.

Keywords: Situation awareness, Ontology-driven information systems

1. Introduction

A common problem in large-scale control systems is, not least because of the increased amount of sensed information, that human operators are at risk to get lost in the induced information overload. This fact entails a lack of awareness of the available information's overall

[☆]This work has been partly funded by the Austrian Federal Ministry of Transport, Innovation and Technology (BMVIT) under grant FIT-IT Semantic Systems no. 819577.

[*]Corresp. author. Tel./Fax: +43 1 512 30 20-0 / -99

Email addresses: `norbert.baumgartner@te-am.net` (Norbert Baumgartner), `gottesheim@tk.jku.at` (Wolfgang Gottesheim), `mitsch@tk.jku.at` (Stefan Mitsch), `werner.retschitzegger@univie.ac.at` (Werner Retschitzegger), `wieland.schwinger@jku.at` (Wieland Schwinger)

meaning, i.e. a lack of *Situation Awareness* (SAW), which hampers the timely and correct resolution as well as pro-active prevention of critical situations.

The steps to achieve situation awareness by humans have already been defined by Endsley about twenty years ago [1]: "the perception of elements in the environment within a volume of time and space, the comprehension of their meaning, and the projection of their status in the near future". To achieve this, the JDL data fusion model provides a widely accepted definition of situation awareness [2] comprising, on the one hand, the estimation of the states of physical objects, and, on the other hand, the estimation of relationships among entities, by means of, e. g., situation assessment algorithms, thereby enabling humans to achieve situation awareness. The actual implementation of this definition in large-scale control systems, however, is still a matter of research struggling for automated techniques beyond the mere presentation of information in a graphical user interface as focused in most existing approaches. To exemplify the complexity of achieving SAW in large-scale control systems, we reflect on the area of road traffic management (RTM) being our demonstration domain throughout the paper.

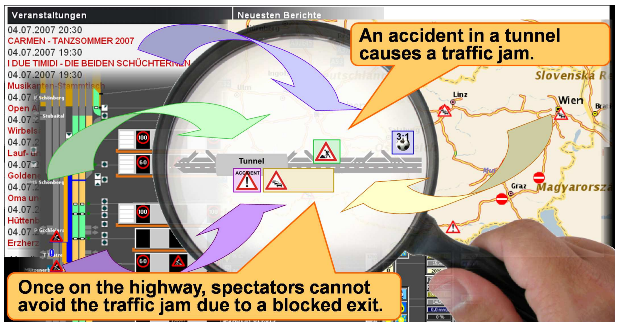
An accident in a tunnel causes a traffic jam.
Once on the highway, spectators cannot avoid the traffic jam due to a blocked exit.
Veranstaltungen
Neuesten Berichte
Tunnel
Linz
Wien
Graz
Slovenská R
Magyarorsz

Figure 1: A critical road traffic situation induced by a lack of SAW.

Let us assume that, as depicted in Fig. 1, an operator of an RTM system observes an accident in a motorway tunnel through a surveillance camera, and due to that, a build-up of a traffic jam. The first reaction according to the operating procedures would be to open the emergency lane within the tunnel, allowing motorists to bypass the accident and avoiding the traffic jam from building up further. Unless consulting a public event information system, the operator might not be aware of the fact that in a short while a football game is about to end and that many spectators will naturally utilize the tunnel for their way home. Although the emergency lane's capacity would not be able to dissolve the traffic jam in this case, spectators could, however, surpass the traffic jam ahead by using a nearby exit. Unfortunately, scheduled roadworks are blocking this exit; a fact the operator is only aware of if consulting the road maintenance time table. Hence, an operator should not only be aware of the option of opening the emergency lane, but also of the non-obvious possibility of canceling the blocking road works before the football fans are stuck in traffic, thus timely avoiding the emergence of a critical situation, i.e., a severe traffic jam.

The scenario demonstrates that awareness of the available information's overall meaning and its implications is of paramount importance for a smooth functioning of the environment under control. Currently, the operator's SAW in such scenarios is de facto scarcely supported by existing RTM systems, not least since the information stem from multiple heterogeneous and unlinked information sources (which is of course not as obvious as the magnifying glass suggests in Fig. 1). Unfortunately, this finding applies to large-scale control systems in general (cf. [3]).

In recent years, ontologies have been regarded to be suitable for developing SAW systems (e. g., [2]). Research, however, has been focused on concrete domains and is hardly generalizable—concrete implementations across application domains such as military operations, road traffic management or even pervasive computing tend to reinvent the wheel. The few existing *domain-independent* ontology-based approaches are still limited regarding their reusability.

In this paper, we present our experience gained from the development of BeAware!, a framework for ontology-driven information systems which aims at increasing an operator's SAW. The linchpin of our framework is a domain-independent SAW core ontology which leverages the reusability of BeAware! by the incorporation of spatio-temporal primitive relations between observed real-world objects. Moreover, the ontology fulfills the vision of an ontology-*driven* information system. It is an integral part of BeAware! throughout its whole architecture—it is used for persisting information, it provides a model for communicating situations to operators and domain experts, it allows us to declaratively integrate information sources, it provides the vocabulary for defining the relevant types of situations, and, finally, offers knowledge that may be exploited in situation assessment algorithms. Therefore, the ontology determines the development of BeAware! at design time and run time as well.

The paper is structured as follows. First, we provide an overview of BeAware! and its constituents, especially the SAW core ontology, in Sect. 2 and Sect. 3. The main features of BeAware!, i.e., information integration, the definition of relevant situation types, and situation assessment are discussed in Sect. 4. The applicability of BeAware! in a real-world setting is shown in Sect. 5 by means of a proof-of-concept prototype in the RTM domain, finally leading to lessons learned for the development of ontology-driven information systems in Sect. 6. The paper is concluded with a discussion of related work in Sect. 7 and an overview of further prospects in Sect. 8.

2. BeAware! in a Nutshell

In this section, we introduce BeAware!'s overall architecture and identify its main components such as the SAW core ontology the framework is driven by. To provide concrete examples, we depict BeAware! in the context of an RTM system by a walk through the main processing chain (cf. Fig. 2).

We begin our walkthrough with the information issued by the information sources on the left-hand side of the architecture. To simplify the discussion, we assume that each information source already provides its information in some domain ontology (e. g., a road maintenance system's ontology). To utilize notions from the field of Description Logics (DL) [4], the domain ontology consists of a T-box (terminological box) defining the domain model and an A-box (assertions box) containing the ontology's individuals such as planned roadworks. As usual in large-scale control systems, the information sources provide their information in an asynchronous manner since most information on the real-world objects of interest are event-triggered. We integrate information using a hybrid approach [5], i. e., we define a separate domain ontology per data source, thereby preserving the semantics of the source, built upon a global shared vocabulary. The mappings we use to relate these ontologies to the original source are defined by domain experts and follow the approach of structure enrichment [5], that is, we define a logical model resembling the structure of the original source that contains additional concepts. By providing an integrated knowledge base, we enable reusing facilities for defining relevant *situation types* and assessing the situations of interest in the *Situation Assessor* component. This component searches

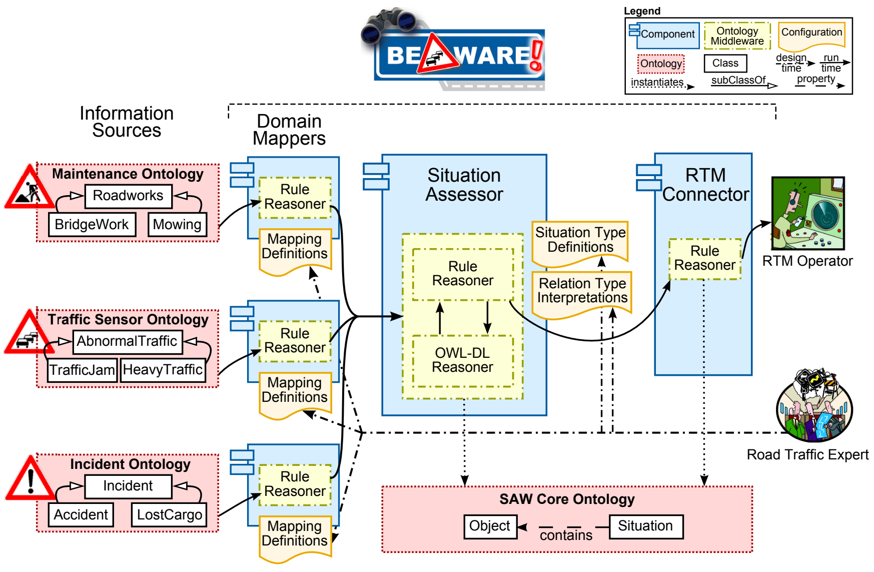

Figure 2: Overall architecture of BeAware!

for *interrelated objects* that match given situation and relation type definitions (e. g., an accident causing a traffic jam). The determined situations, which pinpoint the relevant information, are finally presented via the RTM connector to the RTM operator.

Speaking of situation type definitions, we also have to examine the design time characteristics of BeAware!. The approach resides on the common assumption that a domain expert is in a position to define the situation types of interest. Being an abstract state of affairs (cf. [6]), a situation type is a template for a situation which may be instantiated during *situation assessment*. These situation type definitions, which reuse the vocabulary of the SAW core ontology, constitute, together with domain mapping definitions and interpretations of relations between objects in a concrete domain (e. g., "causes"), the configuration of BeAware!. Note that by the knowledge transfer between the operator and the domain expert, this configuration should be subject to continuing optimization.

The details of the above architecture and our experience gained during its implementation are discussed in the following sections beginning with the SAW core ontology.

3. The SAW Core Ontology

The basic concepts of the SAW core ontology, designed for reusability as suggested in [7] and depicted in Fig. 3 represent the common ground of current approaches (e. g., SAWA by Matheus et al. [8]). The essential difference of our approach is the incorporation of spatio-temporal relation types, which are introduced in the following section. In particular, related approaches, such as the ones by Matheus et al. [9] and Kokar et al. [10], provide only limited support for time and space [11] in the form of time instants on objects (but not pre-defined relation types), and locations (but neither spatial reference frame, nor pre-defined spatial relation types, as proposed in our previous work [12]). Instead, these works require the users of their ontologies to define relation types. Although being very generic, such an approach leads to a number of drawbacks, like, e. g., ontologies and rules being hard to keep consistent [8] and algorithm optimization not being focused on. Moreover, relation individuals in these approaches are not derived from object attributes during situation assessment, but assumed to be already asserted in the ontology.

The basis for deriving relations in our SAW core ontology are individuals of type Object, which describes real-world objects with a spatio-temporal extent (e. g., a traffic jam) by means of attributes (e. g., a location). Situations consist of objects and Relations derived from object attributes. Moreover, situations are themselves integrated as objects into the ontology and can therefore also participate in relations. Because of the necessity to associate relations with properties such as a time interval of validity, they are lifted to the class level; hence, two object properties model the association with objects (cf., relatesFrom and relatesTo). An instance of Role provides further insight into a concrete relation by associating objects with relations (e. g., the role *causer* in the relation *causes*). ThematicRoles are analogous concepts and associate objects with situations thereby determining the role of an object in the context of a concrete situation (e. g., to differentiate between active objects, such as an accident, and passive objects, such as a tunnel). An instance of Event, is associated with its affected instance of Attribute and distinguishes between information updates (e. g., the creation of an object).

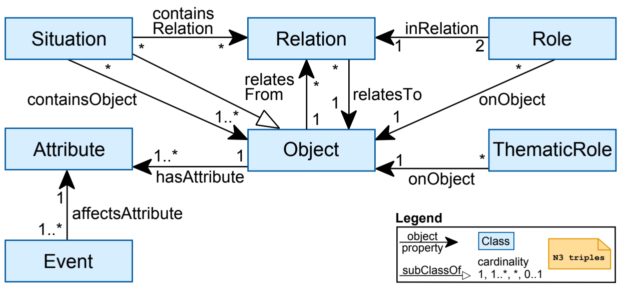

Figure 3: The SAW core ontology.

Note that the SAW core ontology serves as an extension point for domain ontologies. Their domain-dependent concepts extend the core ontology's corresponding base concept. As an example we introduce an RTM ontology: the modeled concepts of the domain focus on typical traffic objects. These are, e. g., RoadWorks, Incident with its specialized concept Accident, Traffic Jam, as well as road weather-related concepts (all are direct or indirect subclasses of the base concept Object).

4. From Information Integration to Situation Assessment

In this section, we discuss the main features of BeAware!—the integration of information by means of the SAW core ontology and the definition of situation types as the configuration for situation assessment.

4.1. Information Integration

The domain-independent SAW core ontology is used for integrating various domain ontologies. To implement a concrete SAW application, BeAware! provides a Domain Mapper component translating between domain ontologies and the SAW core ontology on basis of declaratively defined mappings. Besides simple translations, these mappings have to be capable of extending the SAW core ontology by individuals not yet existing in BeAware!'s A-box—such mappings are known as heterogeneous mappings (e. g., a new traffic jam has to be transfered into BeAware's integrated A-box).

In search for appropriate techniques, we had to discover that most research focuses on ontology matching, e. g., [13], or on so-called homogeneous mappings, such as [14], which are far too simple regarding our requirements. In contrast, we face the problem of heterogeneous mappings. To give an example, a property of an individual in the domain ontology (e. g., the datatype property hasSeverity) should be lifted to the class-level in our SAW core ontology. That is, the mapped property should be represented by an individual belonging to a subclass of Attribute (e. g., an instance of a class Severity). The lifting thus implies the creation of a new class derived from Attribute. Related work—such as Ghidini et al. [15], who proposed extensions to OWL-DL, or Euzenat et al. [16], who presented a future ontology alignment language—dealing with such heterogeneous mappings just recently emerged and is, to the best of our knowledge, not yet sufficiently mature for a stable implementation: instead, we provide a highly specialized and, thus, convenient, mapping vocabulary for the constructs of our SAW core ontology (e. g., containing the translation construct mapsToClass for class-to-class mappings, or the alignment construct mapsToCustomAttribute, which defines a new attribute in the SAW core ontology). For each construct in the domain ontology, a domain expert defines a rule-like RDF statement using one of the mapping constructs from the vocabulary. Thereby integrated information sources benefit from reusing the domain-independent situation assessment facilities described in the following section.

4.2. Situation Type Definition

In this section, we outline our approach to provide a domain expert with a tool for defining the situation types of interest. We describe situations in terms of rule-based situation types comprising objects and the relations between them. The design rationale for our approach is to foster re-use by the exploitation of the SAW core ontology. That is, references to domain ontologies should be minimized in order (i) to provide situation type definitions which are exchangeable across systems and domains and (ii) to develop domain-independent situation assessment facilities as described in Sect. 4.3. For achieving this goal, we introduce the notion of primitive relations.

4.2.1. Primitive Relations

The derivation of appropriate relations among objects is an integral task in situation assessment. In particular, [17] suggests the identification of the types of relations that should be derived as one of the key challenges within the field of SAW. In our previous work [12] we introduced the essential category of *primitive relations*, which are reusable across a wide range of domains and focus on possibly many objects. The practical advantages of directly incorporating such primitive relations into a SAW system are threefold: (i) domain-independent as well as optimized relation derivation algorithms can be developed, (ii) situation types can be defined by explicitly using existing primitive relations, and (iii) the strictly top-down approach, which leads to a restricted deterministic view of relations, may be levered.

In search for relevant families of primitive relations—i.e., relations associating objects according to the same aspect—we follow the definition of SAW by Endsley [18], examining objects "[. . .] within a volume of time and space". Hence, in our previous work [19], we suggested several families of relations modeling different spatio-temporal aspects such as mereotopology, orientation, distance, and size. The chosen families originate from well-known calculi in the field of qualitative spatio-temporal reasoning. Of course, the following listing of potential primitive relations is not complete, but the chosen ones are regarded to be domain-independent and, thus, generally applicable for SAW. Additional families of primitive relations can be integrated to extend BeAware!'s modeling capabilities. Many other families, like the proposed standing relations in [8], are dependent on a particular domain and therefore not reusable as primitive relations. In this paper, we provide a summary of reusable extensions to the SAW core ontology (cf. Fig. 4 which details its concept Relation with different concrete relation families). We exemplarily extend TemporalRelation with FreksaRelation, implemented by the concrete relation type Older. The other packages are integrated likewise, but omitted for brevity. Domain-dependent relations are then defined using primitive relations (e. g., obstructs from RTM could be defined using the spatial relation rcc-8:PO—partly overlapping—and the temporal relation allen:contains) leading to reduced efforts in developing systems achieving SAW.

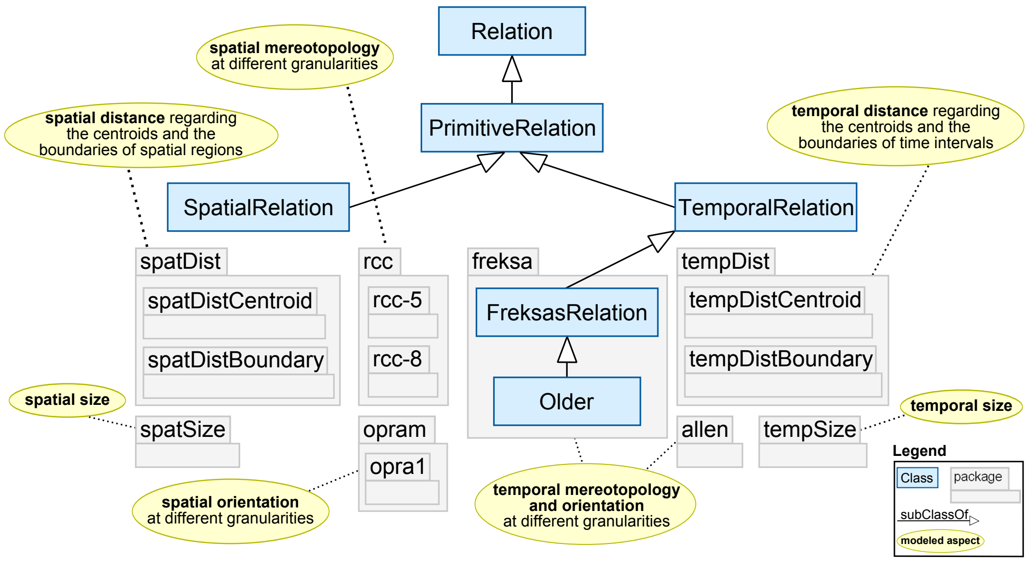
Relation
PrimitiveRelation
SpatialRelation
TemporalRelation
spatial mereotopology at different granularities
spatial distance regarding the centroids and the boundaries of spatial regions
temporal distance regarding the centroids and the boundaries of time intervals
spatDist
spatDistCentroid
spatDistBoundary
rcc
rcc-5
rcc-8
freksa
FreksasRelation
Older
tempDist
tempDistCentroid
tempDistBoundary
spatial size
spatSize
opram
opra1
allen
tempSize
temporal size
spatial orientation at different granularities
temporal mereotopology and orientation at different granularities
Legend
Class
package
subClassOf
modeled aspect

Figure 4: Extension of the SAW core ontology by spatio-temporal relations.

Temporal mereotopology and orientation. Because we mainly deal with time intervals as temporal primitives, we included Allen's well-known time intervals algebra that provides thirteen relations between time intervals covering mereotopology and orientation [20]. An example for such a relation is road works *overlaps* and *in front of* traffic jam (indicates, that road works causing a traffic jam start before the traffic jam occurs, but when they end the traffic jam still exists). Another incorporated, less fine-grained family are Freksa's relations between semi-intervals (e. g. road works *older* traffic jam, [21]).

Spatial mereotopology and orientation. An example for spatial mereotopology is the Region Connection Calculus [22]; especially its version with eight relations (RCC-8) is well known for representing mereotopological relationships between spatial regions. Examples for relations are

X *disconnected* Y (indicates that X and Y are spatially disparate), and X *non-tangential proper part* Y (indicates that X is spatially completely included in Y, and that its boundaries do not meet Y's boundaries, e. g., an accident inside a tunnel). Spatial orientation is covered by the Oriented Point Relations Algebra with different granularities (OPRA$_m$) [23].

Spatial and temporal size and distance. The aspects distance and size are represented by corresponding spatial and temporal families of relation types. For spatial size and distance, such families abstract from the concrete spatial reference frame, since deriving and interpreting such relations is a domain-dependent task. BeAware! interprets size and distance information between centroids and boundaries of object shapes, as we interpret objects not only as points, but as occupying a region. Examples for such relations are road works *close* tunnel (indicates a spatial distance), or traffic jam *shorter* road works (indicates a temporal size). Next, we have a look at the definition of situation types based on the primitive relations introduced above.

4.2.2. Rule-Based Situation Types

In accordance with our previous work [24], we use a simple, rule-based formalism to define situation types based on our SAW core ontology. We elaborate the formalism using the traffic situation type depicted in Fig. 5, denoted as S_0: if the locations of a traffic jam and an area of fog are *partly overlapping* and *very close* to each other, we found an instance of our situation type S_0.

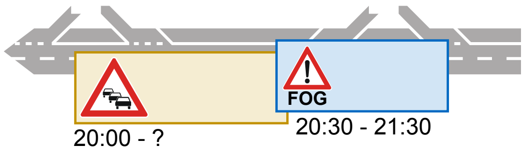

Figure 5: S_0: Fog in the border area of a traffic jam.

Such an informally described situation type is formalized by means of the following rule. For reasons of brevity, we apply a simplified syntax based on the Semantic Web Rule Language (SWRL[1]) for class membership and relation types. For an object variable being a member of the class C, we write C(?o); for two object variables being in concrete relationship defined by the relation type R, we write R(?o1, ?o2). Thus, the above situation type is defined as follows:

S_0: rtm:TrafficJam(?o1). rtm:Fog(?o2).
 rcc-5:PO(?o1,?o2). spatDistBoundary:VeryClose(?o1,?o2).

The situation type specification rule is an implication, i.e., the right-hand side represents the trigger for the instantiation of the situation type S_0. Note that we assume that corresponding interpretations of rcc-5:PO and spatDistBoundary:VeryClose are provided analogously to the definition of the situation type. That is, during situation assessment, object attributes such as the location or a lifespan are examined to instantiate these relations as the basis for aggregating objects to situations (i.e., instantiate a situation type).

The following section discusses how such situation types defined at the design time of a system can be efficiently instantiated at run time.

[1]http://www.w3.org/Submissions/SWRL

4.3. Situation Assessment

In this section, we will examine the exploitation of the ontology introduced above during situation assessment, i.e., the instantiation of rule-based situation types. In detail, starting with an analysis of the complexity of situation assessment, we further on discuss specific reasoning shortcuts to optimize situation assessment and facilitate the knowledge encoded in the SAW core ontology. We finish this section with an overview of the assessment of evolving and just partly matching situations.

4.3.1. The Complexity of Situation Assessment

The assessment of situations can be seen as a satisfiability problem, i.e., checking which object configurations match the defined situation types. We follow a logic programming, backward chaining approach for assessing situations, because of the complexity of relation derivation. In case we implemented a forward chaining approach, we would usually derive at least one relation per relation type for each pair of objects implying a *constant* runtime and space complexity of $O(o^2 r)$, where o is the number of all objects and r is the number of relation types. Since usually not all holding relations among all objects as well as all relation types are necessary, we reduce o as well as r by being more selective—we utilize backward chaining and optimize it by providing shortcuts to previously derived or deduced relations ensuring that each relation is, as with forward chaining, just derived once. In contrast to a constant complexity, we would thereby just face a *worst* case runtime and space complexity of $O(o^2 r)$. The situation assessment algorithm may be sketched as follows. We try to satisfy each situation type by satisfying the contributing object types and relation types. Before trying to directly derive a relation type using the provided, potentially domain-specific interpretation of, for example "partly overlapping", we try to perform some shortcuts for deducing relations from existing ones rather than deriving them from actual attributes. Each thereby discovered relation is, independent from its actual usage in a situation, kept in the ontological knowledge base.

4.3.2. Reasoning Shortcuts

The first reasoning shortcut exploits just the SAW core ontology and aims at the inference of relations. The basic underlying principle is that there are *disjoint*, *equivalent* or *subsumed* primitive relation types, even across families of relations. Disjointness is exploited to avoid comparisons which are a priori known to fail. That is, in case we have derived a relation of type R1 between two objects, we can spare the assessment of all relations that are defined to be disjoint from R1 and omit them in further derivations. Considering that most families of relation types introduced so far are joint exhaustive and pairwise disjoint, we can anticipate a significant reduction of actual object comparisons using this shortcut. Equivalent and subsumed relation types basically express that a derived relation adhering to some relation type is, by definition, also a member of the relation type's equivalent or subsuming relation types. An example is the equivalence of relation types between the RCC families RCC-5 and RCC-8 (rcc-5:PO ≡ rcc-8:PO). Thereby, we can potentially spare the computationally intensive derivation of relations and enable further inferences with respect to situation types such as *satisfiability checking* and *minimization*.

The key to checking the satisfiability of a situation type is to examine its contributing relation types. The satisfiability of two relation types is determined by the subsumption lattice of the corresponding family or, in case the relation types belong to different families, the subsumption relationships in-between. A situation type is satisfiable, if none of its associated relation types are by definition disjoint. Otherwise, the corresponding situation type would be unsatisfiable. An example for an unsatisfiable situation type is

S_a: rtm:Accident(?o1). rtm:Fog(?o2).
rcc-8:EC(?o1,?o2). spatDistBoundary:Far(?o1,?o2).

because rcc-8:EC (externally connected) is a subset of spatDistBoundary:VeryClose and therefore disjoint from spatDistBoundary:Far. A situation type is additionally minimal, if no subsets of its relation types can be replaced with a single equally expressive relation type. Otherwise, the situation type would become simpler with respect to the contributing number of relation types.

The characteristics *symmetry*, *inverseness* and *transitivity* are important meta information about relation types (e. g., if a relation is symmetric, it is no longer necessary to differentiate between the relation's left-hand side and its right-hand side). The usage of *composition tables*, which are closely related to transitive relations, is well established in the field of qualitative spatio-temporal reasoning. A composition table maps two relation types (R1, R2) to potentially multiple relation types (R3). If a relation r1 of type R1 between two objects ?o1 and ?o2 holds as well as a relation r2 between two objects ?o2 and ?o3 then a relation instance of R3 (r3) between the objects ?o1 and ?o3 can be deduced. The performance implications of applying these shortcuts are discussed in Sect. 5.3.

4.3.3. Of Situations and Their Neighbors

Situation assessment, as described above, still faces two inherent challenges: first, operators are interested in evolutions of situations, which is crucial for the early detection of emerging instances of interesting situation types to pro-actively take appropriate actions. Second, operators also want to be informed about situations which are similar to or just partly matching interesting situation types, because sensors still just capture a very limited view of the real world. In this section, based on our previous work [24], [25], we describe an approach exploiting *conceptual neighborhoods* (known, e. g., also in the area of qualitative envisioning [26]) of relations, which, generalized to conceptual neighborhoods of situations, are the basis for addressing both challenges.

The evolution of a situation may be seen as a course of events [6]. We have, however, no a priori knowledge about the evolution of situations and, consequently, can not determine whether a situation may evolve into or is similar to an instance of a most-critical situation type. Therefore, we base on the notion of conceptual neighborhoods of relations to model evolutions of situations. Two relations are, according to Freksa [21], conceptual neighbors, "if a direct transition from one relation to the other can occur upon an arbitrarily small change in the referenced domain". We assume that for each family of relations a directed graph specifying the conceptual neighborhood between its relations in terms of a Conceptual Neighborhood Graph (CNG) is given. Fig. 6 shows an examplary CNG for the RCC-5 relation family.

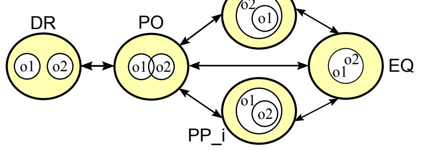

Figure 6: The conceptual neighborhood graph of RCC-5.

The CNG consists of five relations (exemplified with the objects o1 and o2 depicting the relation's meaning) and the possible evolutions in-between. A relation between two objects evolves in the form of single-hop transitions with respect to the CNG of its corresponding family. For example, if the relation DR (discrete from) holds between o1 and o2, it can only evolve to EQ (equals) by traversing over PO (partly overlapping). We further define the direct neighborhood of a situation s as the set of situations containing the same objects as s and only relations that are reachable by a single transition of one of the relations contributing to s. A situation's direct neighborhood includes similar or just partly matching situations. Let us demonstrate the concept using the exemplary situation type S_0, "Fog in the border area of a traffic jam". A situation s_a of type S_a defined as

S_a: rtm:TrafficJam(?o1). rtm:Fog(?o2).
rcc-5:DR(?o1,?o2).

is, based on the CNG of RCC-5, then at least two transitions away from being an instance of S_0 (DR→PO→PP). With this knowledge at hand, we are in a position to determine whether a situation may evolve into a critical one or just fuzzily matches a situation type definition.

To sum up, the described concepts and situation assessment facilities of BeAware! clearly highlight the central role of the SAW core ontology. It facilitates declarative information integration and enables the domain expert to define situation types in a rule-based manner, thereby sticking to his or her familiar domain vocabulary together with domain-independent "glue" in the form of primitive relations from the SAW core ontology. Primitive relations are then exploited during situation assessment in terms of reasoning shortcuts and the assessment of evolving as well as just partly matching situations. In the next section, we describe the application of these approaches in the context of a prototypical implementation in the RTM domain.

5. Prototypical Implementation in the RTM Domain

In this section, we demonstrate the applicability of BeAware! by an overview of our prototypical implementation in the RTM domain. After describing the software infrastructure and the setting of our prototypical implementation, we discuss the evaluation of BeAware! from different aspects.

5.1. Software Infrastructure

BeAware! is largely based on Java and the Jena Semantic Web Framework[2], chosen for its industrial background, the large and supportive developer community, and the clean API. For OWL reasoning within Jena, we use the Pellet OWL Reasoner[3] being smoothly integrated in Jena and offering reasonable performance. For situation assessment, we additionally use the RDF triple store AllegroGraph[4]. AllegroGraph is a client-server system that integrates with Jena and Pellet on the client side. On the server side, AllegroGraph provides a Prolog and Common Lisp environment which both are employed for developing the situation assessment algorithms of BeAware!.

[2]http://jena.sourceforge.net
[3]http://pellet.owldl.com
[4]http://agraph.franz.com/

5.2. BeAware! in a Real-World Setting

The test bed for the prototypical implementation is the real-world traffic information processed within the RTM system of the Austrian highways agency ASFINAG[5], which currently provides traffic information from the following information sources: a roadworks management system, a traffic jam detection system, an incident management system, and a nation-wide radio broadcasting station.

Quantity structure. The majority, that is, about 80%, of the available traffic information is related with roadworks and the corresponding traffic restrictions. The remaining traffic information usually deal with traffic jams and incidents like accidents, or wrong-way drivers. Currently, one has to anticipate a constant load of 250 traffic objects, for which corresponding information is available. The amount of traffic objects to be handled by BeAware! will increase even further if, as planned, scheduled public events and forecast weather information are incorporated. Then, at peak times, one may anticipate a maximum of 350 traffic objects on Austrian highways, which have to be managed by an operator. Regarding the frequency of updates, we observe an average of about three information updates per minute with, however, a relatively high deviation ranging from no updates for a couple of minutes to peaks of about 15 updates per minute. Naturally, most of the updates result from changing traffic jams.

Situation types. In collaboration with the Austrian highways agency, we have defined 10 critical situation types that are relevant for RTM, as shown in Tab. 1. Most of these situation types use a combination of spatial distance and mereotopology (e. g., a traffic jam is obstructed by a closed exit), some additionally employ temporal mereotopology and orientation in case the temporal sequence is important (e. g., an accident causes a traffic jam).

5.3. Evaluation

BeAware! has been evaluated in three ways: manual test cases using a GIS-based graphical user interface, in-depth cases studies of complex traffic situations, and performance test cases with real-world traffic situations.

Manual test cases. The manual test cases have been performed in order to introduce potential users of the system into the concept of situations and to let traffic engineers experiment with different situation type configurations. After a brief introduction into the domain-independent concepts and primitive relations provided by BeAware!, the test users quickly adopted the concepts of modeling situations using rule-based situation types following a top-down approach as proposed in [27]. They were able to represent their domain-specific relations using primitive relations and managed to represent situation types as desired. Although our rule-based approach was therefore shown to be easily applicable, we noticed the need for a visual situation type editor simplifying the choice of appropriate relations to model situation types. Such an editor should support them by presenting relevant objects and relations that might constitute new situtation types based on, for instance, repeated ocurrences, and identify frequent deviations from existing situation type definitions to highlight the need for adjustment.

[5]http://www.asfinag.at

Table 1: Situation types

Description	Situation Type Definition
Poor driving conditions at the fringe of a traffic jam	PoorDrivingConditions(?pdc) ∧ TrafficJam(?j) ∧ PartiallyOverlapping(?pdc, ?j) ∧ VeryClose(?pdc, ?j)
A traffic restricting action (e.g., road works) occurs	Roadworks(?r) ∧ Obstruction(?obs) ∧ Equals(?r, ?obs)
An area of poor driving conditions causes an accident	PoorDrivingConditions(?pdc) ∧ Accident(?a) ∧ (PartiallyOverlapping(?pdc, ?a) ∨ (ProperPart(?a, ?pdc)) ∧ OlderContemporaryOf(?pdc, ?a)
A traffic jam potentially merges with another traffic jam	TrafficJam(?j1) ∧ TrafficJam(?j2) ∧ Disrelated(?j1, ?j2) ∧ Close(?j1, ?j2)
A wrong-way driver heads towards roadworks	WrongWayDriver(?w) ∧ Roadworks(?r) ∧ Disrelated(?w, ?r) ∧ Commensurate(?w, ?r)
A wrong-way driver rushes into roadworks	WrongWayDriver(?wwd) ∧ Roadworks(?r) ∧ ProperPart(?wwd, ?r)
An accident causes a traffic jam	Accident(?a) ∧ TrafficJam(?j) ∧ ExternallyConnected(?a, ?j)∧OlderContemporaryOf?a, ?j)
An accident occurs in the area of roadworks	Accident(?a) ∧ Roadworks(?r) ∧ ProperPart(?a, ?r)
A traffic obstruction occurs near a public event	Obstruction(?obs) ∧ PublicEvent(?pe) ∧ Close(?obs, ?pe) ∧ During(?pe, ?obs)
A traffic jam is obstructed by a closed exit	TrafficJam(?j) ∧ ClosedExit(?clex) ∧ TangentialProperPart(?clex, ?j) ∧ VeryClose(?j, ?clex)

Case studies. The performed case studies have been complex combinations of the 10 critical situation types mentioned above to show the correctness and applicability of the situation assessment algorithms. Details of these case studies focusing on the assessment of evolving and similar situations are presented in our previous work [24]. The main insight, however, is the lack of an appropriate distance measure between two situations. Although we know that one situation may evolve into another one, a realistic metric on how to estimate the probability of such an evolution is an important and still open issue.

Performance tests. We examined the performance of BeAware! from two perspectives. First, we aimed at providing guidelines for the usage of the introduced reasoning shortcuts. The main finding was that the reasoning shortcuts based on subsumption lattices and symmetry are the most beneficial ones, as they impose just a small overhead on the reasoning engines and their return on investment is rather high. The utilization of inverseness has been a mixed blessing, but was still more beneficial than employing transitivity and composition tables. Both shortcuts suffer from the problem that the "hit rate", i.e., the number of inferred relations actually contributing to a situation, is very low in our setting. Thus, the return on investment heavily depends on the actual situation type configuration indicating that an adaptive selection of shortcuts based on the configuration at hand could be promising. The second aim of the performance tests was to evaluate BeAware! in a real-world setting. Utilizing the shortcut based on the relation types' subsumption lattices, we could process a constant number of 500 objects without risking an overflow of the processing chain. Although this result demonstrates the applicability of BeAware! in the RTM domain, it may not be sufficient for other domains. Thus, further work focuses on leveraging the scalability of the situation assessment algorithms.

Investigating reusability. The measures taken for evaluating BeAware! show that it is indeed applicable to the RTM domain and that the introduced approaches successfully solve the described challenges. We now evaluate BeAware!'s reusability from an application developer's perspective. A main advantage for an application developer is that the framework provides out-of-the-box situation assessment facilities whose utilization involves, in the end, the implementation of clear-cut as well as fine-granular pieces of business logic such as the interpretation of primitive relations in a concrete domain and the declarative specification of situation types. The most valuable advantage from our point of view, however, is the flexibility of situation type configurations. Situation types as well as their constituents are not statically included in the ontology and can thus be adapted in a hands-on fashion. For example, the increase of the focus of a situation type by replacing the relation type rcc-5:PO with rcc-8:EC is just a matter of seconds. This flexibility is not just valuable during the development of a system, but also simplifies the operational phase, since feedback from human operators about the performance of the system may be swiftly incorporated—notably always sticking to the vocabulary provided by the aligned SAW core ontologies. Finally, the universality and articulation of the ontology are surely important for the adoption of the framework in a concrete domain. To sum up, universality is obtained with the exception of a universally applicable spatial framework (e. g., a graph network in the case of RTM). Articulation is obtained by pluggable families of relation types from which an application developer may select the ones appropriate for her application.

6. Lessons Learned

In this section, we detail concrete issues we came across during the implementation of BeAware! and which we consider to be crucial for the design of ontology-driven information systems in general.

Missing support for temporal information. The first issue concerns the currently missing support for the *temporalization of information* in ontologies. The developers of SAWA have also outlined their problems with temporal information [28]—they highlighted the general problem of frequent updates to an ontological knowledge base and the performance problems involved when querying temporalized information. Though we also had to deal with these issues when entering and querying individuals, our concerns are rather related with the overall topic of representing temporalized information. If we assume that each indvidual or even concept is tagged with a time interval of validity, it is questionable why such a fundamental kind of information has not found its way into the core of RDF and OWL. From our point of view, handling and querying temporalized information should be standardized on one of the lower layers of the Semantic Web stack such as proposed by Gutierrez et al. [29]. Thereby, efficient reasoning mechanisms could be developed for processing such information. For example, AllegroGraph, the triple store employed in BeAware!, already offers specialized and thus very efficient reasoning facilities for temporal information. In fact, AllegroGraph also provides spatial inferences which, as we think, should be the next step after providing a common set of inferences for the temporal dimension.

Frequent updates to ontology A-boxes. Given the software architecture outlined above, another issue is the tracking of the A-box individuals which are actually affected by frequent and asynchronous updates. We found that this update tracking can only be done *efficiently* on the layer of RDF. In detail, we can reduce the occurring updates to added or removed RDF triples, which we consequently cache during a transaction comprising several updates. Based on this cache, the actually changed individuals are determined upon the end of a transaction and the execution of possible business logic such as situation assessment. For example, we thereby invalidate no longer existing relations between objects as a prerequisite to situation assessment. Another convenient feature of Jena in this respect is the possibility to apply *set-theoretic operations* on ontology models, i.e. A-boxes or T-boxes. For example, the set of triples which are *not* affected by a transaction may easily be determined by removing the set of changed triples from the current A-box.

Integrating Jena, Pellet, and AllegroGraph. In BeAware's overall architecture, Jena integrates different reasoning engines (AllegroGraph as rule reasoner and Pellet as OWL-DL reasoner) and the overall Java application. Reconsidering the internal architecture of the Situation Assessor, the Pellet reasoner is well-integrated into the Jena model. In order to use the T-box as well as A-box inferences of Pellet within AllegroGraph, we followed two approaches. First, we *materialized* the T-box inferences such as subclass relationships from Pellet and transfered them to AllegroGraph. Since this is a one-shot action and AllegroGraph is well-suited for a large number of triples, this approach is rather straight-forward. Second, for transferring A-box inferences to AllegroGraph, we followed a more sophisticated strategy. Since BeAware! operates in a well-known environment, we exactly know the OWL inferences the business logic of the Domain Mapper and the Situation Assessor rely on. The analysis of these constraints has shown that the only features beyond the simple RDFS inferences provided by AllegroGraph are the classification of relation individuals, i.e., the derivation of defined relation types. Since we know, however, that an assessed relation individual initially belongs to exactly one relation type, we incorporate templates for relation individuals per relation type into the A-box and materialize their inferred relation types within Pellet. In case we derive a relation within AllegroGraph, we have a look at the already materialized types of the corresponding template relation. The consequence is that analogous to the transfer of the T-box inferences, the A-box inferences can be transfered to AllegroGraph in a one-shot manner. Pellet is thus just used for the materialization of T-box inferences, the template individuals for relations and their materialized A-box inferences. The effect of this slight deviation from the originally planned more tightly coupled architecture regarding the interplay between the OWL-DL reasoner Pellet and the rule reasoner AllegroGraph is a better overall scalability, since communication between the two reasoning components is minimized.

7. Related Work

We discuss related work in SAW and, especially, situation assessment to provide further insights into the current challenges. Notably, most current research focuses on a concrete application domain. Interesting surveys origin in the domain of military operations, in which situation awareness currently operates on the level of information fusion (e. g., [30],[31], and [32]). Summing up the findings from the above surveys, we discover two main streams of approaches: syntactic approaches based on some kind of logic, and statistical approaches. The first kind of approaches almost completely relies on ontologies: AKTiveSA [33] integrates heterogeneous information sources and rather passive SAW facilities ranging from a GIS-enabled ontology browser over information filtering to semantic queries, but situation assessment is, in contrast to BeAware!, not supported. In [34], Boury-Brisset argues for a goal-driven development of domain-specific ontologies based on the decisions a human operator has to make. Moreover, she recommends to utilize domain-independent ontologies for knowledge reuse. These two propositions reflect the view also followed in BeAware!, that is, to enable an application developer to focus on the goals of an operator in a certain domain by introducing a domain-independent SAW ontology. The statistical approaches (e. g., fuzzy logic [35] and Bayesian belief networks [36]), though mainly working with numerical information, also assume the presence of symbolic information—for example, in the form of a Bayesian belief network's nodes or the fuzzy sets in fuzzy logic. According to Lambert [37] the choice of appropriate symbols is therefore one of the most important challenges of situation awareness, which again may be solved with ontologies.

Another very active research community works in the domain of pervasive computing (see, e. g., [38] for a survey of approaches). Mastrogiovanni et al. [39] describe their architecture using rather generic concepts such as Situation and User, but do not depict a complete domain-independent ontology. They focus on modeling and assessing situations by exploiting the reasoning capabilities of DL with respect to a hierarchy of situations. In contrast to our approach, they implicitely encode relations into situations (e. g., a situation called inBed implies the relation in between a user and a bed), which reduces reusability. Another ontology-based approach from the area of pervasive computing is suggested by Korpipää et al. [40]. The exemplary vocabulary they provide focuses on sensor data and, thus, deals with information fusion rather than with situation awareness. Moreover, their actual ontology is a set of key-value pairs extended by meta-information characterizing a situation (e. g., context type, source, or confidence). Although this approach is certainly generic, it is not very expressive and does not a provide a domain-independent conceptual model for SAW. In summary, current approaches in pervasive computing favor syntactic situation assessment analogously to the military domain, whereby, we also notice a strong tendency towards ontologies.

Motivated by this agreement on ontologies, in our previous work [11] we evaluated existing domain-independent ontologies for their applicability to BeAware!. Most influencing to our work are SAWA [8], SOUPA [41], the situation ontology [42], and CONON [43]. SAWA, since it origins from the field of SAW and not only focuses on pervasive computing, exceeds the other approaches in terms of the concepts incorporated into the ontology, but still misses qualitative approaches to the representation of time and space. SOUPA enables representating time intervals and enhances the ontology by incorporating the Region Connection Calculus, yet it lacks representation of situations and situation types. The situation ontology provides ways to combine situations and treats situations as objects, but its ontology misses SAW-specific concepts such as a representation of space and time and situation types. Thematic roles are incorporated in CONON and SOUPA. Note, however, that none of the evaluated approaches supports situation types, which can be interpreted as a general negligence of concepts for classifying situations. In our BeAware! SAW core ontology, we incorporated the strengths of these approaches and eliminated their weaknesses such as the missing spatio-temporal *primitive* relations. That is, in contrast to the rather straightforward spatio-temporal concepts (e. g., spatiallySubsumedBy [41]) available in existing approaches, we focused on primitive relations originating from well-defined spatio-temporal calculi which are agreed to be comprehensive as well as semantically clear, and enable the out-of-the-box situation assessment facilities stated above.

8. Outlook

This concluding chapter identifies open issues and discusses future research topics. A core question is the interpretation of relation types with respect to different spatial frameworks. Which combination of domain-specific and domain-independent spatial frameworks (e. g., generic graph networks or geographic coordinate systems) is adequate can only be determined in the context of a concrete application. Nevertheless, we may increase the universality of BeAware! by providing further spatial frameworks together with pre-defined relation bindings. Further research towards the combination of such frameworks should substantially leverage their reusability.

Besides the open issues identified in Sect. 5.3, action awareness and awareness maintenance will be addressed as broad research topics in the course of our ongoing work. As an assistant for an operator, BeAware! should propose appropriate actions and highlight the effects actions can incur. We therefore will develop an action awareness core ontology integrated with the SAW core ontology, and appropriate reasoning mechanisms to assess actions. The elaboration of such action assessment functionality thereby also has to take into account non-trivial scenarios, which, e. g., even deal with situations that may not have emerged to the point in time, but are likely to do in the future. In this sense action assessment is the basis for pro-actively making decisions to positively influence the evolution of situations.

Since not all situations that may possibly occur in BeAware!'s usage can be foreseen, we will support the maintenance of both situation and action awareness. A process to keep awareness representations up-to-date has to developed, and mechanisms to automatically generate situation types based on a single situation occurrence have to be provided. Finally, to improve action awareness support, actions both taken and merely assessed have to be monitored to prioritize actions for future assessmments with this information.

References

[1] M. R. Endsley, Design and evaluation for situation awareness enhancement, in: Proc. of the Human Factors Society 32nd Annual Meeting, Human Factors Society, Santa Monica, CA, USA, 1988, pp. 97–101.

[2] J. Llinas, C. Bowman, G. Rogova, A. Steinberg, Revisiting the JDL data fusion model II, in: Proc. of the 7th Intl. Conf. on Information Fusion, 2004, pp. 1218–1230.

[3] N. Baumgartner, BeAware! — an ontology-driven framework for situation awareness applications, Ph.D. thesis, Johannes Kepler University Linz (October 2008).

[4] F. Baader, The Description Logic Handbook: Theory, Implementaion and Applications, Press Syndicate of The University of Cambdrige, 2003.

[5] H. Wache, T. Vögele, U. Visser, H. Stuckenschmidt, G. Schuster, H. Neumann, S. Hübner, Ontology-based integration of information — a survey of existing approaches, in: Proc. of the 17th Intl. Joint Conf. on AI, Workshop: Ontologies and Information Sharing, 2001, pp. 108–117.

[6] J. Barwise, J. Perry, Situations and Attitudes, MIT Press, 1983.

[7] E. Simperl, Reusing ontologies on the semantic web: A feasibility study, Data and Knowledge Engineering 68 (10) (2009) 905–925.

[8] C. Matheus, M. Kokar, K. Baclawski, J. Letkowski, C. Call, M. Hinman, J. Salerno, D. Boulware, SAWA: An assistant for higher-level fusion and situation awareness, in: Proc. of SPIE Conf. on Multisensor, Multisource Information Fusion: Arch., Algorithms, and App., 2005, pp. 75–85.

[9] C. J. Matheus, M. M. Kokar, K. Baclawski, A core ontology for situation awareness, in: Proc. of the 6th Intl. Conf. on Information Fusion, 2003, pp. 545–552.

[10] M. M. Kokar, C. J. Matheus, K. Baclawski, Ontology-based situation awareness, Journal of Information Fusion 10 (1) (2009) 83–98.

[11] N. Baumgartner, W. Retschitzegger, A survey of upper ontologies for situation awareness, in: Proc. of the 4th Intl. Conf. on Knowledge Sharing and Collaborative Engineering, ACTA Press, 2006, pp. 1–9.

[12] N. Baumgartner, W. Retschitzegger, Towards a situation awareness framework based on primitive relations, in: Conf. Proc. of 2007 Information, Decision, and Control, IEEE, 2007, pp. 291–295.

[13] G. Pirro, D. Talia, UFOme: An ontology mapping system with strategy prediction capabilities, Data and Knowledge Engineering 69 (5) (2010) 444–471.

[14] J. Euzenat, An API for ontology alignment, in: Proc. of the 3rd Intl. Semantic Web Conf., Springer, 2004, pp. 698–712.

[15] C. Ghidini, L. Serafini, S. Tessaris, On relating heterogeneous elements from different ontologies, in: Proc. of the 6th Intl. and Interdisciplinary Conf. on Modeling and Using Context, Springer, 2007, pp. 234–247.

[16] F. Scharffe, J. Euzenat, A. Polleres, Processing ontology alignments with sparql (position paper), in: Proc. of the 2nd Intl. Conf. on Complex, Intelligent and Software Intensive Systems, Intl. Workshop on Ontology Alignment and Visualization, IEEE, 2008, pp. 913–917.

[17] M. M. Kokar, Situation awareness: Issues and challenges, in: Proc. of the 7th Intl. Conf. on Information Fusion, 2004, pp. 533–534.

[18] M. Endsley, Situation Awareness Analysis and Measurement, Lawrence Erlbaum Associates, New Jersey, USA, 2000, Ch. Theoretical Underpinnings of Situation Awareness: A Critical Review, pp. 3–33.

[19] N. Baumgartner, W. Retschitzegger, W. Schwinger, Lost in time, space, and meaning—an ontology-based approach to road traffic situation awareness, in: Proc. of the 3rd Workshop on Context Awareness for Proactive Systems, 2007.

[20] J. F. Allen, Maintaining knowledge about temporal intervals, Communications of the ACM 26 (11) (1983) 832–843.

[21] C. Freksa, Temporal reasoning based on semi-intervals, Artificial Intelligence 54 (1) (1992) 199–227.

[22] A. G. Cohn, S. M. Hazarika, Qualitative spatial representation and reasoning: An overview., Fundamenta Informaticae 46 (1-2) (2001) 1–29.

[23] R. Moratz, F. Dylla, L. Frommberger, A relative orientation algebra with adjustable granularity, in: Proc. of the Workshop on Agents in Real-Time and Dynamic Environments, 2005.

[24] N. Baumgartner, W. Retschitzegger, W. Schwinger, G. Kotsis, C. Schwietering, Of situations and their neighbors—evolution and similarity in ontology-based approaches to situation awareness, in: Proc. of the 6th Intl. and Interdisciplinary Conf. on Modeling and Using Context, Springer, 2007, pp. 29–42.

[25] N. Baumgartner, W. Gottesheim, S. Mitsch, W. Retschitzegger, W. Schwinger, Situation prediction nets—playing the token game for ontology-driven situation awareness, in: Proceedings of the 29th International Conference on Conceptual Modeling, Springer, Vancouver, Canada, 2010.

[26] E. Davis, to appear in: Qualitative Spatio-Temporal Representation and Reasoning: Trends and Future Directions, Information Science Pub., 2010, Ch. Qualitative Reasoning and Spatio-Temporal Continuity.

[27] J. Holsopple, S. J. Yang, Designing a data fusion system using a top-down approach, in: Proc. of IEEE MILCOM, 2009.

[28] C. J. Matheus, Using ontology-based rules for situation awareness and information fusion, in: Proc. of the W3C Workshop on Rule Languages for Interoperability 2005, W3C, 2005.

[29] C. Gutierrez, C. A. Hurtado, A. Vaisman, Introducing time into rdf, IEEE Transactions on Knowledge and Data Engineering 19 (2007) 207–218.

[30] D. L. Hall, J. Llinas, Handbook of Multisensor Data Fusion, CRC Press, 2001, Ch. Multisensor Data Fusion, pp. 18–27.

[31] J. Biermann, Some experiences with experimental high level fusion systems, in: Proc. of the 10th Intl. Conf. on Inf. Fusion, IEEE, 2007.

[32] K. Sycara, R. Glintona, B. Yu, J. Giampapa, S. Owens, M. Lewis, C. Grindle, An integrated approach to high-level information fusion, Information Fusion 10 (1) (2009) 25–50.

[33] P. R. Smart, A. Russell, N. R. Shadbolt, M. C. Schraefel, L. A. Carr, AKTiveSA: A technical demonstrator system for enhanced situation awareness in military operations other than war, The Computer Journal 50 (6) (2007) 703–716.

[34] A.-C. Boury-Brisset, Ontological engineering for threat evaluation and weapon assignment: a goal-driven approach, in: Proc. of the 10th Intl. Conf. on Information Fusion, IEEE, 2007.

[35] L. A. Zadeh, Fuzzy sets, Information and Control 8 (3) (1965) 338–353.

[36] J. Pearl, Fusion, propagation, and structuring in belief networks, Artificial Intelligence 29 (3) (1986) 241–288.

[37] D. A. Lambert, A blueprint for higher-level fusion systems, Information Fusion 10 (1).

[38] M. Baldauf, S. Dustdar, F. Rosenberg, A survey on context-aware systems, Intl. Journal of Ad Hoc and Ubiquitous Computing, forthcoming 2 (4) (2007) 263–277.

[39] F. Mastrogiovanni, A. Sgorbissa, R. Zaccaria, A distributed architecture for symbolic data fusion, in: Proc. of the 20th Intl. Joint Conf. on Artificial Intelligence, AAAI Press, 2007, pp. 2153–2158.

[40] P. Korpipää, J. Mäntyjärvi, An ontology for mobile device sensor-based context awareness, in: Proc. of the 4th Intl. and Interdisciplinary Conf. on Modeling and Using Context, Springer, 2003, pp. 451–458.

[41] H. Chen, T. Finin, A. Joshi, Ontologies for Agents: Theory and Experiences, Whitestein Series in Software Agent Technologies, Springer, London, 2005, Ch. The SOUPA Ontology for Pervasive Computing.

[42] S. S. Yau, J. Liu, Hierarchical situation modeling and reasoning for pervasive computing, in: Proc. of the 3rd Workshop on Software Technologies for Future Embedded and Ubiquitous Systems, 2006, pp. 5–10.

[43] X. H. Wang, D. Q. Zhang, T. Gu, H. K. Pung, Ontology based context modeling and reasoning using OWL, in: Proc. of the 2nd Annual Conf. on Pervasive Computing and Communications, IEEE, 2004, p. 18.