*In Proceedings of Sixth International Conference on Information Fusion, pages 545-552, Cairns, Australia, July 2003.*

# A Core Ontology for Situation Awareness

**Christopher J. Matheus**
Versatile Information Systems, Inc.
Framingham, MA, USA
cmatheus@vistology.com

**Mieczyslaw M. Kokar**
Northeastern University,
Boston, MA, USA
kokar@coe.neu.edu

**Kenneth Baclawski**
Northeastern University,
Boston, MA, USA
ken@baclawski.com

**Abstract -** *In this paper we present an ontology for situation awareness. One of our goals is to support the claim that this ontology is a reasonable candidate for representing various scenarios of situation awareness. Towards this aim we provide an explanation of the meaning of this ontology, show its expressiveness and demonstrate its extensibility. We also compare the expressiveness of this ontology with alternative approaches we considered during the design of the ontology. We then show how the ontology can be adapted to handle domain-specific situations by readily extending the core language. The extensions include adding subclasses, sub-properties and additional attributes to the core ontology. We conclude with an example of how the ontology can be used to annotate specific instances of a situation.*

**Keywords:** Situation awareness, ontology, formal method, information fusion, relation derivation, events.

## 1 Introduction

Maintaining a coherent *situation awareness* (SAW) concerning all units operating in a region of interest (*e.g.*, battlefield environment, emergency disaster scene, counter-terrorism event) is essential for achieving successful resolution of an evolving situation. The process of achieving SAW is called *situation analysis*. The primary basis for SAW is knowledge of the objects within the region of interest, typically provided by "sensors" (both mechanical and human) that perform object identification and characterization -- in military parlance these are known as *level 1* sensors [1]. Although knowledge of the individual objects and their current attributes is essential, this does not by itself constitute complete "awareness" – SAW also requires knowing the relations among the objects that are relevant to the current operation. For example, simply knowing that there is a friendly tank and an enemy tank on the battlefield may not be as important as knowing that the enemy tank is "in firing range" of the friendly tank.

Systems that assist in situation analysis require the ability to represent objects and maintain information about their attributes and relationships with other objects as they evolve over time. This necessitates a model (or more formally, a *theory*) of how the world of the situation "works" in the eyes of those doing the analysis. Such a model can be partially defined by an *ontology* that describes a set of entities (concrete and/or abstract) and the relationships they can have with each other [2]. Clearly, different classes of situations will necessitate different ontologies so as to appropriately define the various objects and relations relevant to their specific domains. We have constructed a core ontology for SAW that provides a basis from which to build ontologies for arbitrary situations. At the center of this model are objects, relations and events. The relationships between these core entities are defined such that a system based on this ontology will be able to represent and capture sufficient information about a situation to support high-level reasoning, which is a primary goal of our research [3].

The development of the ontology proceeded over many months and involved consideration of several alternative approaches. The most challenging aspects of the design revolved around the problem of representing values of attributes and relations that evolve over time and space. In this paper we relate some of these issues and discuss the advantages and disadvantages of various alternative approaches.

In the next section we provide some general background information on SAW, including our formal definition of SAW. This leads into a description of the core SAW ontology followed by a discussion of some important design decisions. To demonstrate how the ontology can be extended to specific domains we introduce a simple battlefield scenario and show how it can be accommodated through the sub-classing of a small number of core SAW classes. Finally, we provide an example of the use of this domain-specialized ontology to describe a specific scenario using the DARPA Agent Markup Language (DAML).

## 2 Situation Awareness

A number of philosophers and logicians introduced concepts similar to that of a situation, including von Mises [4] in 1949 and Bunge [5] in the 1970s. However, the earliest formal notion of *situation* (although not situation awareness) was introduced by Barwise as a means of giving a more realistic formal semantics for speech acts than what was then available [6]. In contrast with a "world" which determines the value of every proposition, a situation corresponds to the limited parts of reality we perceive, reason about, and live in. With limited

information a situation can provide answers to some but not all questions about the world. Furthermore, in situation semantics, basic properties, relations, events and even situations are reified (i.e., made concrete) as objects to be reasoned about [7]. Note that once a situation is made into a concrete object, various properties can be associated with the situation. While Barwise's situation semantics is only one of the many alternative semantic frameworks currently available, its basic themes have been incorporated into most others.

The specific term *situation awareness* is most commonly used by the Human-Computer Interaction (HCI) community (*cf.,* Endsley and Garland [8]). The concerns of this community are to design computer interfaces so that a human operator can achieve SAW in a timely fashion. From this point of view, SAW occurs in the mind of the operator. In almost any fairly complex system, such as military aircraft and nuclear reactors, manual tasks are being replaced by automated functions. However, human operators are still responsible for managing SAW. This raises new kinds of problems due to human limitations in maintaining SAW. The SAW literature gives many examples of incidents and accidents, which could have been avoided if operators had recognized the situation in time.

Situation awareness is also used in the data fusion community where it has been more commonly referred to as *situation assessment*. Data fusion is an increasingly important element of diverse military and commercial systems. The process of data fusion uses overlapping information to detect, identify and track relevant objects in a region. The term "data fusion" is used because information originates from multiple sources. More succinctly, data fusion is the process of combining data to refine state estimates and predictions [1].

| Fusion Level | Association Process | Estimation | Entity Estimation |
|---|---|---|---|
| L.0 Sub-Object Assessment L.1 Object Assessment | Assignment | Detection Attribution | Signal Physical Object |
| L.2 Situation Assessment L.3 Impact Assessment | Aggregation | Relation Plan Interaction | Aggregation Effect (Situation given Plans) |
| L.4 Process Refinement | Planning | (Control) | (Action) |

Table 1 JDL's 5 Levels of Data Fusion

The terminology of data fusion has been standardized by the Joint Directors of Laboratories (JDL) in the form of a so-called JDL Data Fusion Model. In this model, data fusion is divided into five levels as shown in Table 1. Note that SAW is Level 2 data fusion in this model. The JDL model defines SAW to be the "estimation and prediction of relations among entities, to include force structure and cross force relations, communications and perceptual influences, physical context, etc." Level 2 processing typically "involves associating tracks (*i.e.,* hypothesized entities) into aggregations. The state of the aggregate is represented as a network of relations among its elements. We admit any variety of relations to be considered -- physical, organizational, informational, perceptual -- as appropriate to the given need." The table and all quotations in this paragraph are from [1].

Level 3 captures the functionality of the impact assessment, i.e., the effect of actions that are, in part, the result of the processing at the lower level. Level 4 deals with the assignment of resources to tasks. Blasch (*cf.,* [9]) proposed to add one more level – User Refinement – to this model. The goal of this additional level is to make an explicit connection between the computer processing (fusion) of information and human-in-the-loop. The role of the human-in-the loop was also considered in the model proposed in Kokar *et al* [10].

In our research we make use of elements of all three of the frameworks mentioned above (*i.e.,* Logic, HCI and JDL), although we emphasize the terminology and point of view of the JDL model. We favor a formal approach to the problem, as our ultimate intent is to be able to formally reason about situations. Towards this end we have developed a formal definition of SAW, which provides the basis for the rest of the work described in this paper. When we say "formal", we mean an approach in which specifications are first completely expressed in the language of logic and mathematics and then progressively refined by some truth-preserving refinement operations (*cf.,* [11]), using in the process such tools as Specware [12] and theorem provers like SNARK [13].

**Definition:** Situation Awareness (SAW) is knowledge of the following:

- A specification of the Goal theory, $\mathbf{T}_g$;
- An ontology, i.e. a theory $\mathbf{T_O}$ of the world;
- A stream of measurements $\mathbf{W_1}$, $\mathbf{W_2}$… for time instances $\mathbf{t_1}$, $\mathbf{t_2}$,…;
- At each time instance, the fused theory $T^t = \nabla_T (\mathbf{T_1^t}, \mathbf{T_2^t}, \ldots \mathbf{T_n^t})$ that combines all the theories that are relevant to the Goal $\mathbf{T}_g$ as well as the fused theory $\mathcal{T}^{t+1} = \nabla_T (\mathbf{T_1^{t+1}}, \mathbf{T_2^{t+1}}, \ldots \mathbf{T_n^{t+1}})$ that combines all the theories that are relevant to the Goal $\mathbf{T}_g$ at some time $\mathbf{t + 1}$ in the future;
- At each time instance t, the fused model $\mathcal{M}^t = \nabla_M (\mathbf{M_1^t}_{.1}, \mathbf{M_1^t}_{.2}, \ldots, \mathbf{M_2^t}_{.1}, \mathbf{M_2^t}_{.2}, \ldots)$ that combines all models relevant to the Goal $T_g$ as well as the fused model $\mathcal{M}^{t+1}$ at some time t + 1 in the future; and,
- Relations $\mathcal{R}^t \subset \mathbf{O}^t \times \mathbf{O}^t$ relevant at time t, as well as at t+1, $\mathcal{R}^{t+1} \subset \mathbf{O}^{t+1} \times \mathbf{O}^{t+1}$ among objects (here we

consider only binary relations, but the formalization can be extended to include relations of higher arity).

Our core SAW ontology described in the next section effectively defines the "theory of the world", $\mathbf{T_O}$. It contains classes to support all of the formal symbols in the definition (although not always in a 1-to-1 manner).

# 3 The Core SAW Ontology

An ontology is a specification of concepts and relationships among the concepts that can exist in a given setting (*cf.,* [2], [14]). Ontologies were part of the culture in philosophy and linguistics for many years. Then the computer science community in general, and the agents community in particular, started using this concept as a basis for communication among agents (*cf.,* [14]), Recently, ontologies began being used in the information fusion community (*cf.,* [15], [16], [17], [18]).

In our development of a formal approach to reasoning about situations (see [3]) we needed an ontology that would satisfy several requirements. First it needed to be able to represent objects and relationships as well as their evolution over time. Second, we wanted it to be able to express essentially any "reasonable" evolution of objects and relationships (although possibly only approximately). Third, the design needed to be economical so as to ultimately permit its implementation in a working system.

Figure 1 depicts the main portion of the SAW ontology we developed to satisfy these requirement. It is depicted as a UML diagram [19], where rectangles represent classes and connecting lines indicate inter-class relationships. The ***Situation***[*] class (upper right corner) defines a situation to be a collection of *Goals, SituationObjects* and *Relations.* ***SituationObjects*** are entities in a situation -- both physical and abstract -- that can have characteristics (*i.e., Attributes*) and can participate in relationships. ***Attributes*** define values of specific object characteristics, such as weight or color. A ***PhysicalObject*** is a special type of *SituationObject* that necessarily has the attributes of *Volume*, *Position* and *Velocity*. ***Relations*** define the relationships between ordered sets of *SituationObjects*. For example, inRangeOf(X,Y) might be a *Relation* representing the circumstance when one *PhysicalObject*, X, is within firing range of a second *PhysicalObject*, Y.

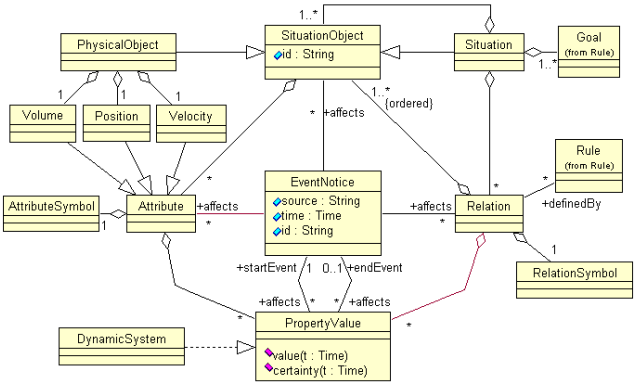


Figure 1. Core SAW Ontology

An important aspect of *Attributes* and *Relations* is that they need to be associated with values that can change over time. To accomplish this *Attributes/Relations* are associated with zero or more ***PropertyValues*** each of which defines two time dependant functions, one for the actual *value* and the other for the *certainty* assigned to that value. A new *PropertyValue* is created for an *Attribute/Relation* whenever an *EventNotice* arrives that "affects" that *Attribute/Relation.* The value of an *Attribute/Relation* at a particular point in time (either current, past or future) can be determined by accessing the value function of the *PropertyValue* instance that is in effect at the prescribed time. This is illustrated in the diagram in Figure 2, but before explaining the illustration we need to introduce the *EventNotice* class.

It is now that we need to introduce the notion of *EventNotices.* ***EventNotices*** contain information about events in the real-world situation observed by a sensory *source* at a specific *time* that *affects* a specific *Relation* or *Attribute* (of a specific *SituationObject)* by defining or constraining its *PropertyValue.* These are the entities that indicate change in the situation and thus are the vehicles by which changes are affected in the *Attributes* and *Relations* of the situation representation.

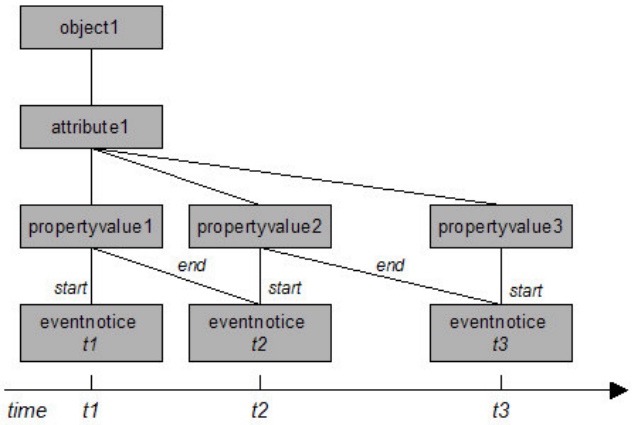


Figure 2. *PropertyValues* delineated by *EventNotices*

[*] We use the convention of capitalizing and italicizing names that refer to classes in the ontology. When we are defining a class we will also make it bold.

Consider now the example depicted in Figure 2. Some event happens at time *t1* resulting in the generation of *eventnotice-t1* by some sensor. This *EventNotice* affects *attrbute1* or *object1* by assigning it a value and certainty instantiated as *propertyvalue1*. At time *t2* a second event occurs generating *eventnotice2* in turn affects *attribute1*, this case by assigning it a new value and certainty in the form of *propertyvalue2*. *eventnotice2* also becomes associated with *propertyvalue1* as it effectively marks the end of *propertyvalue1's* period of being in effect. A similar process occurs with the onset of the third event at time *t3*.

The ontology permits an *PropertyValue* to be implemented as as a model of a *DynamicSystem* that provides a value function parameterized over time; this feature becomes important when a system needs to be able to predict the evolution of *PropertyValues* into the future (see discussion below). Similarly, the *certainty* of a *PropertyValue* can be dynamically modeled in the *DynamicSystem*. For example, the certainty of a value might decay as time goes on in the absence of new observations that affect it.

To illustrate the need for a *DynamicSystem* implementation of *PropertyValues*, consider the *Position* attribute of a *PhysicalObject*. The *Position* attribute is interesting in that its value for an object at time t+1 is related to the *Velocity* (a vector providing speed and direction) of that object at time t. Even if no new *EventNotice* affecting the position is received at time t+1, it is reasonable to assume that the object's position has changed. In the absence of additional information (*e.g.*, acceleration, trajectory) it might be reasonable to assume that the object continues to move at its last noted speed and direction until informed otherwise, all be it with increasing uncertainty as time goes on. To be able to make such projections in the absence of explicit sensory information requires predictive models. It is for this reason that the SAW ontology shows *DynamicSystems* as a way of *implementing PropertyValues*. Certain attributes, such as *Position*, would be modeled by dynamic systems that might themselves generate internal *EventNotices* to update the attribute values, with some lesser degree of certainty, until new external sensory information arrives. It might also become desirable to fuse multiple model-predicted values or to combine model-generated values with sensory information in cases where the certainty of the external information is less than perfect.

## 3.1 Alternative Designs

We now discuss some alternative design approaches considered in the development of our SAW ontology. All of these are concerned in some way with the issue of representing relations and attributes that evolve over time. This issue proved to be the most challenging, in part because of the various ways it can be approached, and in part because of the critical role it must play in a real-world SAW solutions, thus necessitating a solid design.

The first design considered involves what we call a "snapshot" approach because, like a photo snapshot, the entire state of all relations and attributes are captured at a particular instant in time. As shown in Figure 3 the Snapshot class has a *time* property which must be assigned a unique value corresponding to the time the snapshot was taken. Each *Snapshot* contains an aggregation of *AttributeValues* (one for each *Attribute* in the current situation state) and *Relations* (one for each relation that holds to be true at the time of the snapshot).

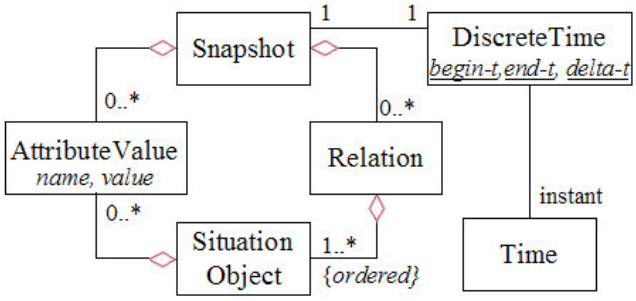


Figure 3. Shapshot Design Alternative

One advantage of this approach is that it is very easy to determine the exact values and relations that hold at any point in time for which a snapshot was taken. The disadvantage is that if a value or relation doesn't change between snapshots you still need to consume the resources necessary to represent the redundant information. For anything but trivially small situation this approach becomes prohibitively expensive for a practical, (near) real-time system. Furthermore, this approach assumes your sensory information comes to you in a lock-step fashion such that all information from all sensors is updated at the same time. This is unlikely to be true in the real world particularly if your sensory information comes from a combination of electro-mechanical sensors and human observers.

Clearly it would be advantageous for it to be possible for sensory information to be updated at a rate that is appropriate for the sensory source and the sensed target. For example, information received about a jet fighter's position from a electronic radar system might need to be recorded in micro-second time intervals, whereas the location of a minefield reported by human observers certainly requires far, far less frequent updating. We therefore want to be able to represent attribute values and relations in such a way they can be updated as frequently or infrequently as necessary. The way we originally proposed doing this is by defining a TimeInterval class that captures the start and end times over which an attribute value or relation holds. In this approach, shown in Figure 4, *Attribute Values* and *Relations* are shown to be associated with aggregations of *TimeIntervals*. In the case of *Relations,* these *TimeIntervals* demark the periods of time for which a relation holds true. For *Attribute Values*

they indicate the time periods for which an attribute's *Value* has a specific value. This approach achieves the effect we were looking for of being able to capture changing values/relations at arbitrary rates and without redundancy. We will tweak it a bit later but first let's will consider another real-world concern.

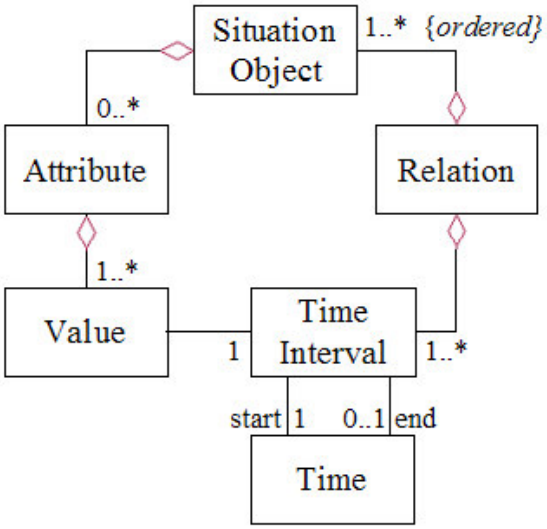


Figure 4. TimeInterval-based Values and Relations

In real-world situations sensory information is not always accurate. To account for this there needs to be a way to represent the certainty/uncertainty inherent in sensory data; this becomes particularly important if the system using the data intends to perform data fusion or higher-order reasoning, as is the case for our solutions. It seems natural to associate certainty/uncertainty with the *Values* of an *Attribute*, but where should this information go when representing *Relations*? Such information does not logically belong with the *TimeInterval* of the *Relation* but if we associate it with the *Relation* itself then the level of certainty/uncertainty is separate from any notion of time and thus must be constant, which clearly isn't accurate. We remedy this problem by changing the *AttributeValue* class into a *PropertyValue* that can also be used by *Relations*. In this way *Relations* are associated with aggregations of *PropertyValues* rather than of *TimeIntervals*, as shown in Figure 6. Now we can add *certainty* to the *PropertyValue* class and have it work for both *Attribute* and *Relations* in the same way. Note that we are not making any claims here about the form of the certainty values that need be used. In our work we have thus far used fuzzy logic to represent certainty and have plans to implement probabilistic and Dempster-Shafer models as well.

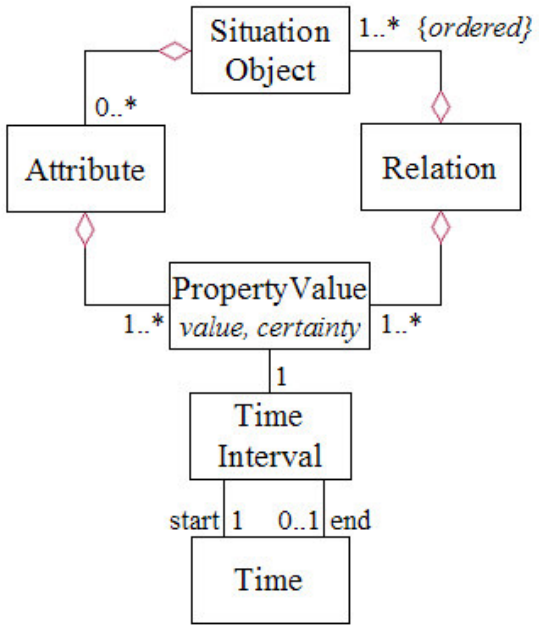


Figure 6. PropertyValues with Certainty

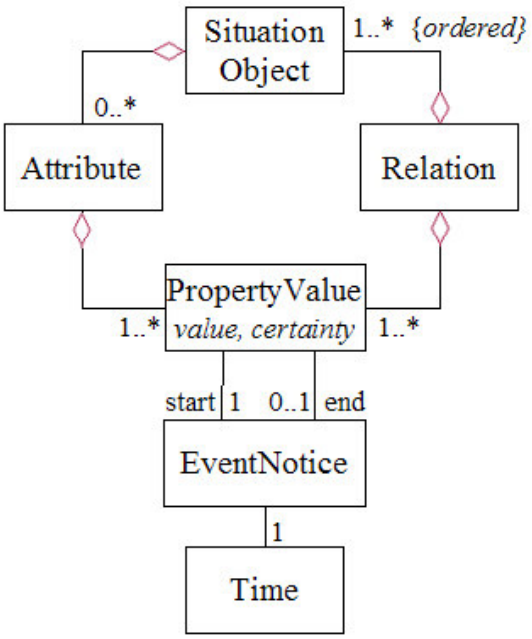


Figure 7. EventNotices as TimeInterval Markers

The final design consideration brings us back to the issue of demarking the beginnings and ends of *PropertyValues*. In the real world, *events*[†] happen which cause (level 1) sensors to transmit new information; we call these notifications of new sensor information *event notices*. Since the onset of these events delineate the times when new values are taken on by attributes and relations it makes sense to use them as the basis for the time intervals

[†] We would like to thank John Salerno of AFRL for the original suggestion to consider the implication of events.

of *PropertyValues*, thereby eliminating the TimeInterval class completely. In the design shown in *EventNotices* are used to mark the start and end times of *PropertyValues*, as was described with an example Section 2 using Figure 2.

# 4 Domain-specific Extensions

The core SAW ontology was designed to be readily extended to support domain specific needs. In this section we present a simple Battlefield scenario and show how the SAW ontology was extended by sub-classing a small number of core classes.

## 4.1 Battlefield Scenario

The Battlefield scenario used in our example consists of two simple snapshots of events describing the initial interaction between two opposing tank platoons (see Figure 9 and Figure 10). In the first snapshot we observe a collection of three stationary blue tanks (tank1, tank2, and tank3), an observation post and a minefield. In the next snapshot (Figure 10), two red tanks (tank4 and tank5) appear approaching from the west and the three blue tanks begin advancing towards them.

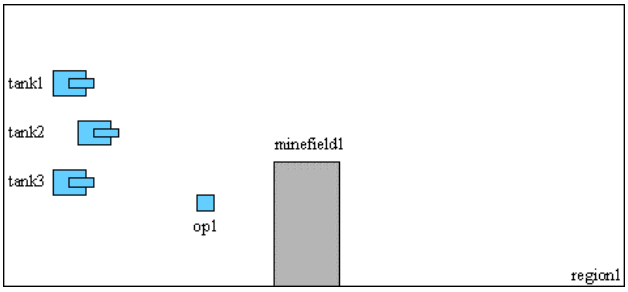


Figure 9. Battlefield Scenario Snapshot 1

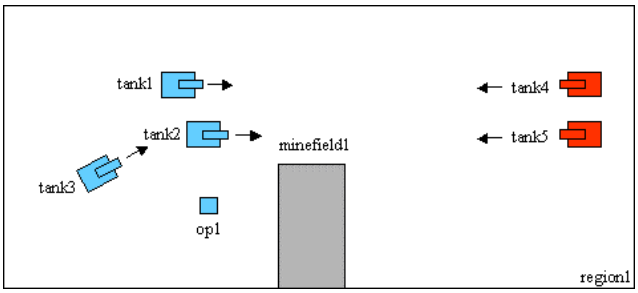


Figure 10. Battlefield Scenario Snapshot 2

To satisfactorily represent the objects in the scenario it was necessary to extend the *PhysicalObject* class to accommodate various military units (see Figure 11) and battlefield obstacles (see Figure 12) (note these are not intended to be complete). The battlefield ontology also creates subclasses of the *SituationObject* to define the abstract notions of doctrines and factions.

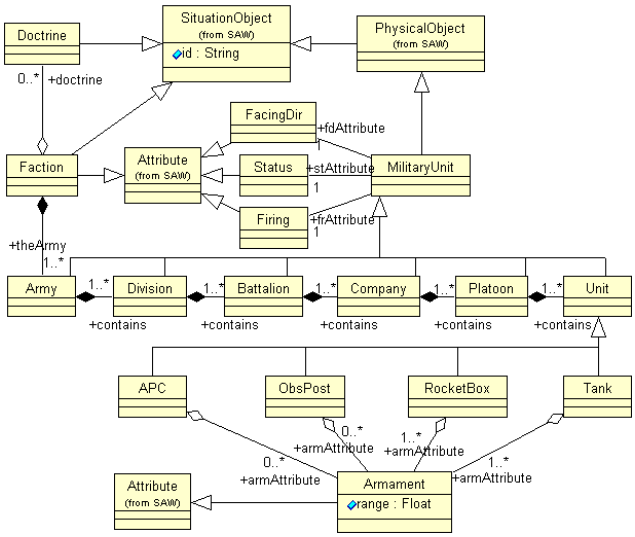


Figure 11. Battlefield Ontology – Military Units

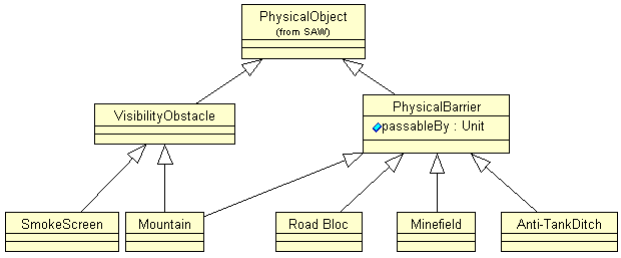


Figure 12. Battlefield Ontology - Obstacles

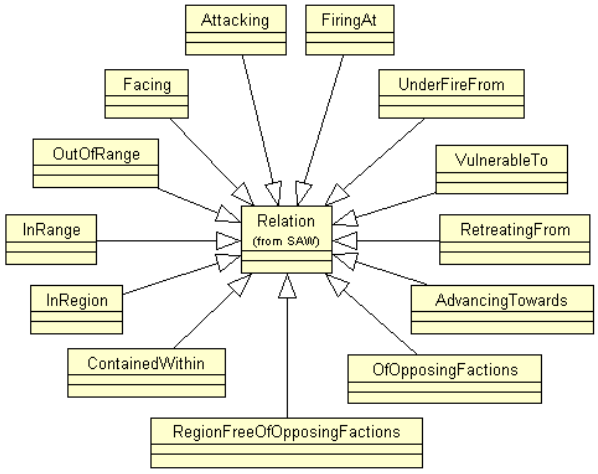


Figure 13. Battlefield Ontology – Relations

In addition, sub-classes of *Relations* specific to the battlefield were added as shown in Figure 13 (again, this is not intended to be a complete list of relevant relations). These *Relations* are important in our system (see [3]) because the intent of our system is to reason about level 1

events in order to determine which level 2 *Relations* are in effect at any given moment.

To complete the ontology for a specific implementation it is also necessary to define subclasses for the *Attributes* of *PhysicalObjects* in order to define how they would be represented in the system. For example, *Position* might be defined to be a two-dimensional vector for a situation where elevation is not a factor.

### 4.2 An Instance Annotation

We now provide an example of how the ontology can be used to create an "instance annotation" of a specific situation using DAML. This example illustrates a partial state of the situation shown in Snapshot 2 (Figure 10). Specifically it shows the values of the *Position* attribute for *tank1* along with one of the *EventNotices* that affected the latest value.

```
<?xml version="1.0"?>
<rdf:RDF . . .>

<saw:PhysicalObject rdf:ID="Tank1">
 <saw:attribute>
  <saw:Position>
   <saw:PropertyValue>
    <saw:PropertyValue>
     <saw:startEvent
       rdf:datatype="xsd:IDREF">
      E1
     </saw:startEvent>
     <saw:endEvent
       rdf:datatype="xsd:IDREF">
      E6
     </saw:startEvent>
     <saw:value rdf:datatype="saw:Vector">
      20,-40
     </saw:value>
     <saw:certainty
       rdf:datatype="xsd:float">
      1.0
     </saw:certainty>
    </saw:PropertyValue>
   </saw:PropertyValue>
   <saw:PropertyValue>
    <saw:PropertyValue>
     <saw:startEvent
       rdf:datatype="xsd:IDREF">
      E6
     </saw:startEvent>
     <saw:value rdf:datatype="saw:Vector">
      70,-40
     </saw:value>
     <saw:certainty
       rdf:datatype="xsd:float">
      1.0
     </saw:certainty>
    </saw:PropertyValue>
   </saw:attributeValuse>
  </saw:Position>
 </saw:attribute>
      ... more attributes ...
</saw:PhysicalObject>

  ... more PhysicalObjects ...

<saw:EventNotice rdf:ID="E6">
 <saw:time rdf:datatype="xsd:dateTime">
  12:10
 </saw:time>
 <saw:eventSource rdf:datatype="xsd:IDREF">
  Sensor1
  </saw:eventSource>
</saw :EventNotice>

      ... more EventNotices ...

</rdf:RDF>
```

*Tank1* is a *PhysicalObject* with a number of attributes including its *Position*. This *Position* attribute is shown in the markup to have two values that are instantiated as *PropertyValue* instances. The first *PropertyValue* instance shows *tank1's* position from the time of *EventNotice* E1 until the time of *EventNotice* E6 to have been the vector coordinates of "20,-40" with certainty of 1.0. The second *PropertyValue* shows the new *Position* of Tank1 reported by *EventNotice* E6, which will stay in affect until another, *EventNotice* occurs that changes it (note the absence of a endEvent value in this *PropertyValue*).

## 5 Conclusions

One of the main objectives of our research in the area of ontologies and information fusion is to develop an approach to situation awareness in which a situation awareness system is flexible enough to accommodate various scenarios of the interaction with the end user. More specifically, the goal is to allow the end user of a SAW system to formulate queries regarding current, and possibly future, situations using an expressive query language. The SAW system then needs to maintain all the necessary information in a well organized fashion to make the answering of the quries possible and efficient. Towards this goal we defined a core ontology. In this paper we showed some design considerations of such an ontology. Our objective was to show that the design decisions we made regarding our core SAW ontology are well justified and rational. In particular, we showed four possible design choices along with advatages and shortcomings of each.

## Acknowledgements

This research was partially supported by AFRL/IF under an SBIR Phase I contract, number F30602-02-C-0039. We would also like to thank Mike Hinman and John Salerno for their helpful suggestions and feedback.

## References

[1] A. Steinberg, C. Bowman, and F. White, "Revisions to the JDL data fusion model", In Proceedings of SPIE Conf. Sensor Fusion: Architectures, Algorithms and Applications III, volume 3719, pages 430-441, April 1999.

[2] Guarino, N., Formal Ontology in Information Systems, Guarino, N. (Ed.), Proc. of Formal Ontology in Information Systems, IOS Press, pages 3-15, June 1998.

[3] C. J. Matheus, K. Baclawski and M. M. Kokar, Derivation of ontological relations using formal methods in a situation awareness scenario, In Proceedings of SPIE Conference on Mulitsensor, Multisource Information Fusion, pages 298-309, April 2003.

[4] L. von Mises, *Human Action: A Treatise on Economics*, originally published in 1949, Fox & Wilkes, 1997.

[5] M. Bunge, *Treatise on basic philosophy. III: Ontology: The furniture of the world*, Reidel, Dordrecht, 1977.

[6] J. Barwise, "Scenes and other situations", *J. Philosophy* 77**,** 369-397, 1981.

[7] J. Barwise, *The Situation In Logic*, CSLI Lecture Notes 17, 1989.

[8] M. Endsley and D. Garland, *Situation Awareness, Analysis and Measurement,* Lawrence Erlbaum Associates, Publishers, Mahway, New Jersey, 2000.

[9 ] Blasch, E. P. and Plano, S. "Level 5: User Refinement to aid the Fusion Process", In Proceedings of SPIE Conference on Mulitsensor, Multisource Information Fusion, pages 288- 297, April 2003.

[10] M. M. Kokar, M. D. Bedworth, and K. B. Frankel. A reference model for data fusion systems. In Proceedings of SPIE Conference on Sensor Fusion: Architectures, Algorithms, and Applications IV, pages 191-202, July 2000.

[11] Formal methods specification and verification guidebook for software and computer systems. Technical Report NASA-GB-002-95, National Aeronautics and Space Administration, 1995.

[12] Specware: Language manual, version 2.0.3. Technical report, Kestrel Institute, 1998.

[13] SNARK: SRI's new automated reasoning kit, 2002. http://www.ai.sri.com/ stickel/snark.html.

[14] D. McGuinness. Ontologies and online commerce. IEEE Intelligent Systems, 16(1):8—14, 2001.

[15] A. C. Boury-Brisset. Towards a knowledge server to support the situation analysis process. In Proceedings of Fusion 2001, 4-th International Conference on Information Fusion, August 2001.

[16] D. A. Lambert. Situations for situation awareness. In Proceedings of Fusion 2001, 4-th International Conference on Information Fusion, August 2001.

[17] J. Roy. From data fusion to situation awareness. In Proceedings of Fusion 2001, 4-th International Conference on Information Fusion, August 2001.

[18] M. M. Kokar and J. Wang. An example of using ontologies and symbolic information in automatic target recognition. In Proceedings of SPIE Conference on Sensor Fusion: Architectures, Algorithms, and Applications VI, pages 40-50. April 2002.

[19] G. Booch ,I. Jacobsen, and J. Rumbaugh. OMG Unified Modeling Language Specification, March 2000.