In Proceedings of SPIE Conference on Multi-sensor, Multi-source Information Fusion, pages 298-309, Orlando, April 2003.

Derivation of ontological relations using formal methods in a situation awareness scenario

Christopher J. Matheus*[a], Kenneth Baclawski[b], Mieczyslaw M. Kokar[b]
[a]Versatile Information Systems, Inc., Framingham, MA
[b]Northeastern University, Boston, MA

ABSTRACT

This paper describes a case study of relation derivation within the context of situation awareness. First we present a scenario in which inputs are supplied by a simulated Level 1 system. The inputs are events annotated with terms from an ontology for situation awareness. This ontology contains concepts used to represent and reason about situations. The ontology and the annotations of events are represented in DAML and Rule-ML and then systematically translated to a formal method language called MetaSlang. Having all information expressed in a formal method language allows us to use a theorem prover, SNARK, to prove that a given relationship among the Level 1 objects holds (or that it does not hold). The paper shows a proof of concept that relation derivation in situation awareness can be done within a formal framework. It also identifies bottlenecks associated with this approach, such as the issue of the large number of potential relations that may have to be considered by the theorem prover. The paper discusses ways of resolving this as well as other problems identified in this study.

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 the situation. The process of achieving SAW is called *situation analysis*. The primary basis for SAW is a knowledge of the objects within the region of interest, specifically object identification and characterization. Considerable effort has been expended on this problem by the Data Fusion community, and numerous hardware and software solutions are at hand.

Although knowledge of the objects is essential, this does not, by itself, constitute complete SAW. It is also necessary to know the relations among the objects that are relevant to the current operation. For example, simply knowing that there is a blue faction tank and a red faction tank on the battlefield may not be as important as knowing that the red tank is in-firing-range of the blue tank. In many ways the problem of finding *all relevant relations* is a more difficult problem than merely determining the objects and their characteristics. While the number of objects may be large, the number scales linearly with the cardinality of the objects in the region. The same cannot be said for relations, whose possibilities increase exponentially as the number of objects increases. Deriving all relevant relations in a situation is at least NP-hard in the number of situation objects.

Our work is concerned not only with developing effective methods for deriving relevant relations from a representation of a situation. We are equally concerned with doing this in a formal manner whereby we can ensure that the derivations are correct, given the soundness of the ground information provided to the process. In this paper we report results from a preliminary case study on the use of a formal methodology for relation derivation applied to a battlefield scenario. We begin with an overview of SAW and a formal definition. This leads into the development of our SAW ontology and our methodology for deriving higher-order relations. We then introduce a battlefield ontology that is built on top of the SAW ontology and use it in a simple scenario which provides the context for our demonstration of the derivation of higher-order relations.

* Contact information: vis@matheus.com.

2. SITUATION AWARENESS OVERVIEW

A number of philosophers and logicians introduced concepts similar to that of a situation, including von Mises[1] in 1949 and Bunge[2] 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[3]. 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. A situation will determine answers in some cases, but not all. Furthermore, in situation semantics, basic properties, relations, events and even situations are reified as objects to be reasoned about[4]. 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[5]). 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 is 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[6].

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 [6] .

In our formalization we will make use of elements of all three of the frameworks mentioned above (*i.e.,* Logic, HCI and JDL), although we will emphasize the terminology and point of view of the JDL model. Before moving on to discuss our SAW ontology and methodology we provide our formal definition of SAW:

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

- *A specification of the Goal theory,* $\boldsymbol{T_g}$*;*
- *An ontology, i.e. a theory* $\boldsymbol{T_O}$ *of the world;*
- *A stream of measurements* $\boldsymbol{W_1, W_2}$*... for time instances* $\boldsymbol{t_1, t_2}$*,…;*
- *At each time instance, the fused theory* $\mathcal{T}^t = \nabla_T (T^t_1, T^t_2, \ldots T^t_n)$ *that combines all the theories that are relevant to the Goal* $\boldsymbol{T_g}$ *as well as the fused theory* $\mathcal{T}^{t+1} = \nabla_T (T^{t+1}_1, T^{t+1}_2, \ldots T^{t+1}_n)$ *that combines all the theories that are relevant to the Goal* $\boldsymbol{T_g}$ *at some time* $\boldsymbol{t+1}$ *in the future;*
- *A each time instance* $\boldsymbol{t}$*, the fused model* $\mathscr{M}^t = \nabla_M (M^t_{1.1}, M^t_{1.2}, \ldots, M^t_{2.1}, M^t_{2.2}, \ldots)$ *that combines all models relevant to the Goal* $\boldsymbol{T_g}$ *as well as the fused model* $\mathscr{M}^{t+1}$ *at some time* $\boldsymbol{t+1}$ *in the future; and,*
- *Relations* $\mathscr{R}^t \subset \boldsymbol{O}^t \times \boldsymbol{O}^t$ *relevant at time t, as well as at t+1,* $\mathscr{R}^{t+1} \subset \boldsymbol{O}^{t+1} \times \boldsymbol{O}^{t+1}$ *among objects (here we consider only binary relations, but the formalization can be extended to include relations of higher arity.*

This is the definition that formed the basis for our formalism of SAW. Further details about this formalism and its realization in Specware can be found in[7,8,9,10]. A more detailed explanation of our formalism and its application to two domains are written up in an Air Force SBIR final report which we intend to turn into additional publications in the near future.

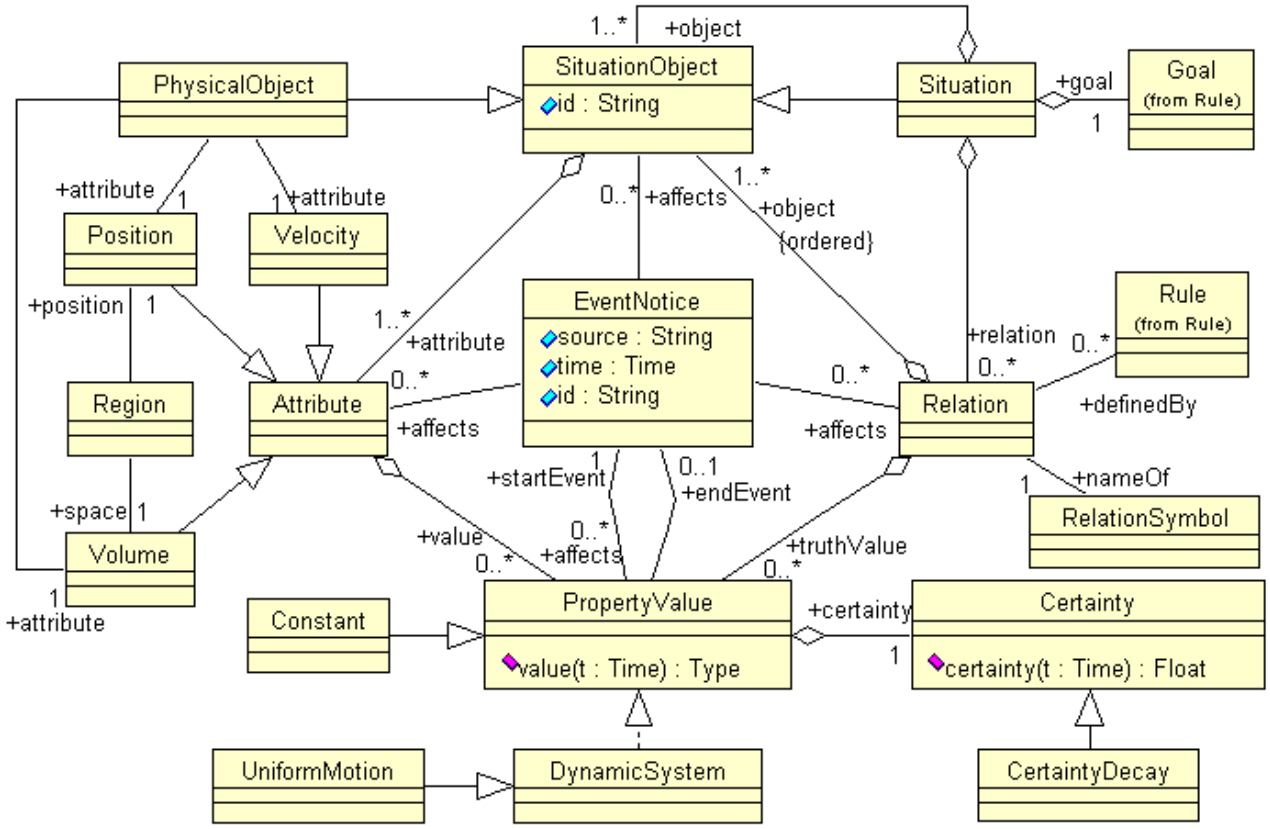

Figure 1 SAW High-Level Ontology

3. SAW ONTOLOGY

We have developed a preliminary SAW ontology, motivated by our formal definition, which defines classes and relations critical to SAW. Figure 1 depicts part of this ontology in the form of a UML diagram. Space prohibits elaboration of all aspects of the ontology, so we limit our explanation to the following primary classes: *SituationObject, PhysicalObject, Attribute, Relation, PropertyValue* and *EventNotice*. 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.

SituationObjects are entities in a situation that can have characteristics (*i.e., Attributes*) and can participate in *Relations*. ***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 values (in this case truth values) of relationships between ordered sets of *SituationObjects*, such as inRangeOf(X,Y). Relations are generally derived by the system although it is possible for them to be reported by external observations.

The values of attributes and relations are defined by a common class called *PropertyValue.* A ***PropertyValue*** provides a time dependent value function over some specific time interval. The time interval is defined relative to a *startEvent* and (possibly) a *stopEvent* that are associated with specific *EventNotices*; Figure 2 demonstrates this with a graphic example. Note that an *Attribute* or *Relation* can, and generally will, have multiple *PropertyValues* because one is generally created with each new sensory observation (*i.e., EventNotice*) concerning the value. ***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.* The ontology permits a *PropertyValue* to be implemented as either a constant or a model of a dynamic system that provides a value function parameterized over time. *PropertyValues* are also associated with a degree of *Certainty,* which is a function over time; the implication is that the certainty of a value decays as time goes on in the absence of new observations that affect it.

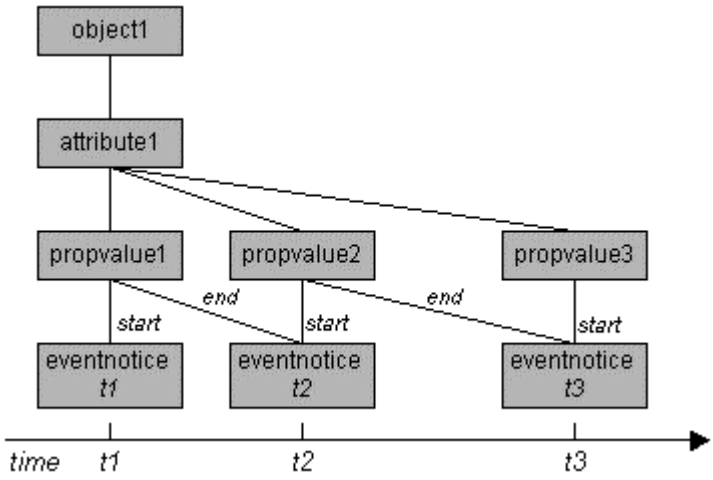

Figure 2 Example of EventNotices delineating PropertyValues

The final critical item in the SAW ontology is that every situation must have a *Goal*. The ***Goal*** defines the objective of the situation and is important because it provides us with a handle on what is *relevant*. Later we will see how *Goals* are related to the *Rules* that define the meaning of *Relations*.

The SAW ontology was first created as a UML diagram and then converted into a DAML ontology using the DAML-UML Enhanced Tool (DUET[11]), the output of which was processed with an XML Stylesheet Language Transform (XSLT) script to make the ontology more human readable. Another XSLT script converts information about classes and relationships into partial MetaSlang *specs* as the initial step in our formalization of SAW.

4. FORMAL METHODLOGY

The methodology we are advocating for formally reasoning about SAW is shown in Figure 3 with individual processes numbered. The rectangles in the diagram denote information representations and the ovals represent processes that convert from one form of information to the next. The grayed ovals are processes that can be fully automated while the other processes require human participation for now and for the foreseeable future. The role of the human in these latter processes, however, can be greatly facilitated through the development of more intelligent tools, some of which we discuss in section 7. We have not yet automated all of the processes that can be automated, which is why we are presenting merely a proof-of-concept at this stage.

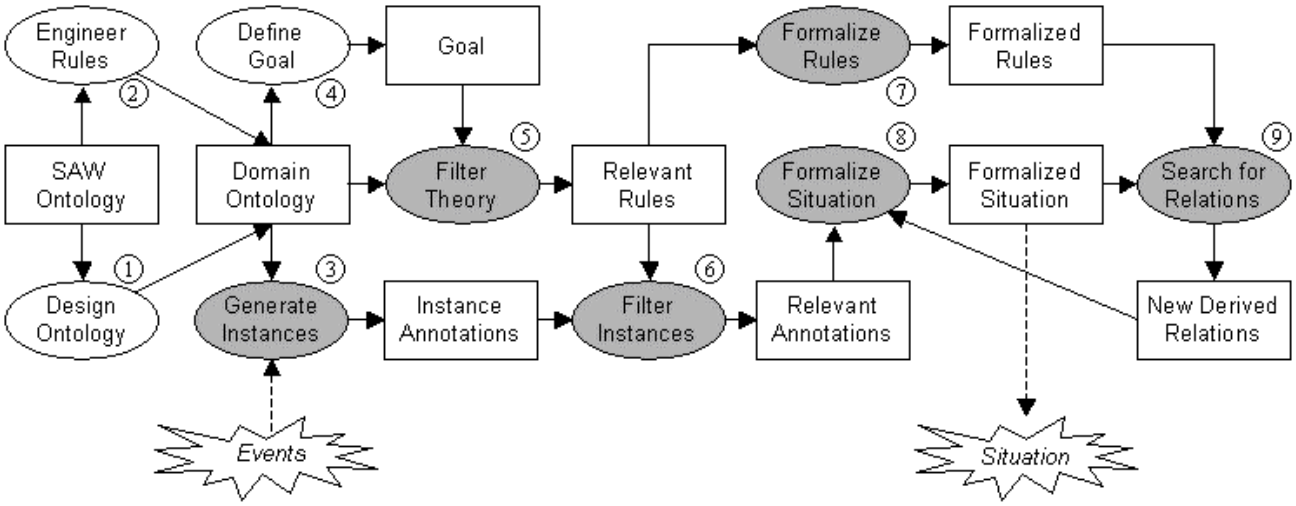

Figure 3 Methodology for Formally Reasoning about SAW

At the root of all processing is our SAW ontology that describes the minimal set of classes and the important relations between them. Domain-specific ontologies are next built on top of the SAW ontology (step 1) by sub-classing these primary classes. In addition, domain-specific rules are developed (step 2) that logically describe the conditions under which each of the sub-classed *Relations* holds true; these rules collectively are called a *theory for the domain*. The new domain ontology forms the language used in describing specific instances of a situation (step 3), which are called *instance annotations*. The ontologies and instance annotations are ultimately converted into formal specifications (steps 7 and 8) that can be processed by a theorem prover (step 9) that derives higher-order relations present in the situation.

As we mentioned earlier, it would be infeasible to search for all possible relations in a situation. We can, however, mitigate this problem if we require each situation to have a goal that is representable as a relation (or set of relations) in the theory for the domain. In this case the only *relevant* rules (and the relations they define) in the situation are those that can participate in a derivation of a proof tree for the goal. The same is true for instances of objects in the situation. With a goal in hand (step 4) we can select relevant portions of the domain theory upfront (step 5) and use this information to focus solely on the relevant situation instances (step 6) and rules/relations.

5. BATTLEFIELD DOMAIN ONTOLOGY

The high-level Battlefield ontology comprises a Military Unit ontology (Figure 4), a Battlefield Relation ontology (Figure 5), and an Obstacle ontology (Figure 6). The ontologies shown are only partial because they are intended for use as a proof-of-concept; a real-world implementation would include many additional sub-classes and relations. Notice how the Military Unit ontology is built on top of the SAW ontology by sub-classing three primary classes *SituationObject*, *PhysicalObject* and *Attribute*. The Obstacle ontology does likewise while the Battlefield Relation ontology sub-classes

the fourth primary class, *Relation*. The three Battlefield ontologies could have as easily been defined as a single monolithic ontology, and in fact, once they are formalized they behave as one. We created them separately because UML diagrams can become unwieldy with so many classes.

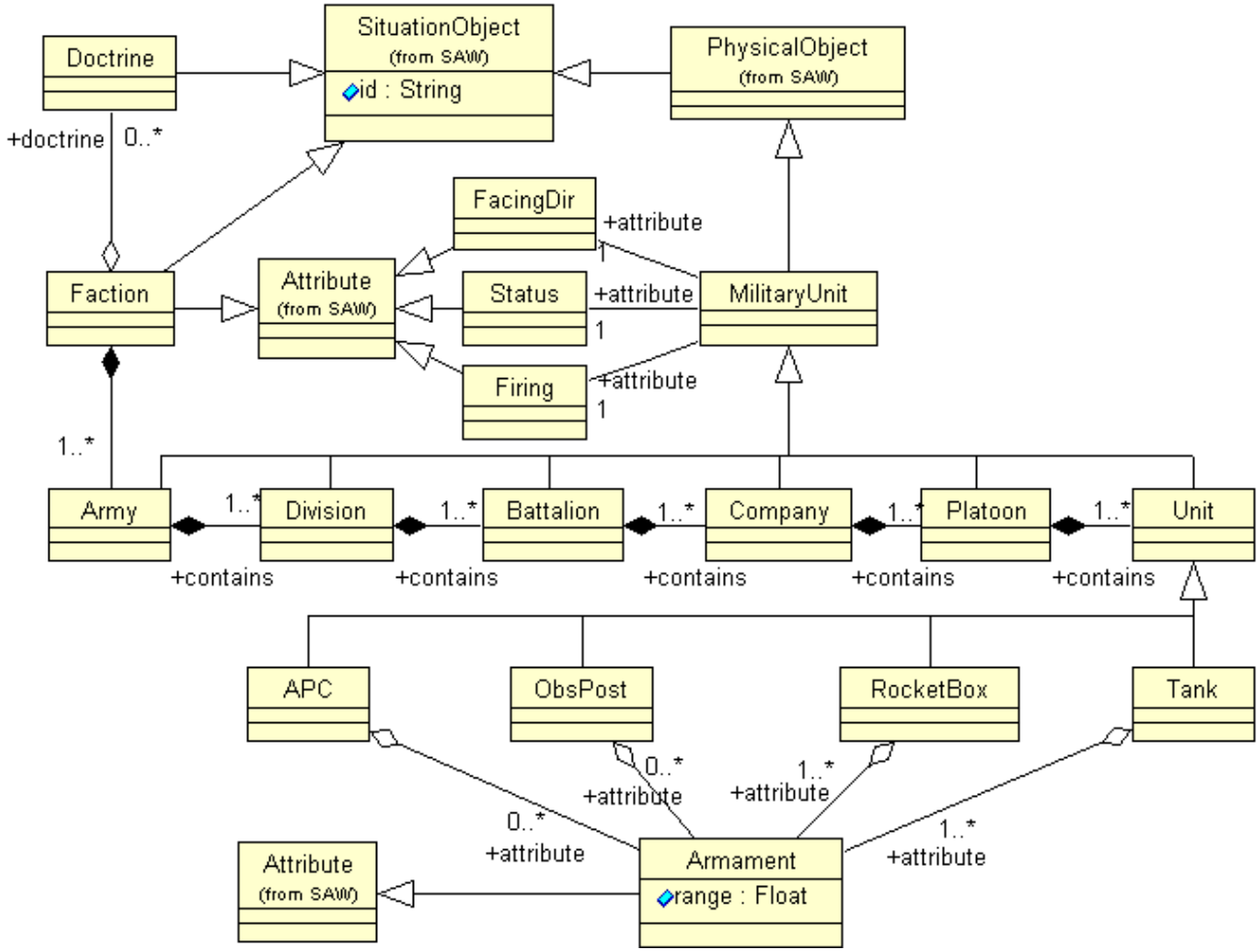

Figure 4 Military Unit Ontology (partial realization)

To complete the Battlefield ontology we need to add rules to define the conditions under which each *Relation* holds true. These rules comprise our battlefield domain theory. For this example we have only developed rules for a subset of the relations depicted in Figure 5. This subset necessarily included those rules that could participate in a derivation tree for the goal of the situation. The goal we chose to use for this example is `defend(MU,R)`, which defines the conditions under which it is true that a military unit `MU` is defending a region `R`. A sample of the rules in English and Rule-ML (an emerging standard for representing rules in XML) is provided in Table 2. With rules represented in Rule-ML it was straightforward to use XSLT to generate a new theory comprised of just the rules relevant to the goal relation (step 5 in Figure 3). Similarly we are able to extract the names of the relevant attributes which can be used to filter the instance annotations and discard irrelevant information (step 6 in Figure 3).

To be able to reason about situations in this domain using Specware and SNARK, we need to have the rules in the form of MetaSlang axioms. We would like to automate the translation from Rule-ML rules to MetaSlang using XSLT, but to do this correctly the rules need to contain information about the sort types of the variables, which is something we have yet to do. Instead, for this example, we simply converted the rules into axioms by hand.

attacking: A unit is attacking another if it is either firing at it or advancing towards its position.

```
<imp name=”Attacking 1”>
 <_head><atom>
   <_opr><rel>attacking</rel></_opr><var>X</var> <var>Y</var>
 </atom></_head>
 <_body><and>
   <atom>
    <_opr><rel>firingAt</rel></_opr><var>X</var> <var>Y</var>
   </atom>
 </and></_body>
</imp>

<imp name=”Attacking 2”>
 <_head><atom>
   <_opr><rel>attacking</rel></_opr> <var>X</var> <var>Y</var>
 </atom></_head>
 <_body><and>
   <atom>
    <_opr><rel>ofOpposingFactions</rel></_opr> <var>X</var> <var>Y</var>
   </atom>
   <atom>
    <_opr><rel>outOfRange</rel></_opr> <var>X</var> <var>Y</var>
   </atom>
   <atom>
    <_opr><rel>advancingTowards</rel></_opr> <var>X</var> <var>Y</var>
   </atom>
 </and></_body>
</imp>
```

firingAt: A unit is firing at an object if the unit is facing the object, the object is in range of the firing unit's armament and the firing unit is indeed firing.

```
<imp name="Firing At">
 <_head><atom>
   <_opr><rel>firingAt</rel></_opr> <var>X</var> <var>Y</var>
 </atom></_head>
 <_body><and>
   <atom>
    <_opr><rel>facing</rel></_opr> <var>X</var> <var>Y</var>
   </atom>
   <atom>
    <_opr><rel>inRangeOf</rel></_opr> <var>X</var> <var>Y</var>
   </atom>
   <atom>
    <_opr><rel>attributeEquals</rel></_opr> <var>X</var> <var>firing</var>
<var>true</var>
   </atom>
 </and></_body>
</imp>
```

Table 2 Sample of Relation Rules in English and Rule-ML

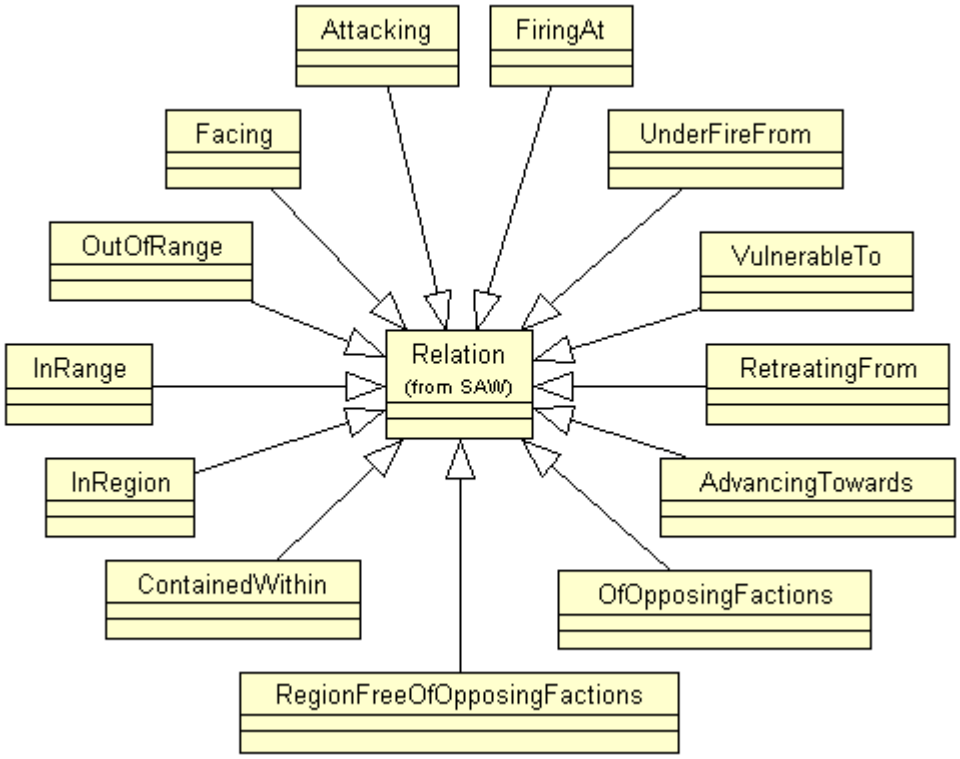

Figure 5 Battlefield Relations Ontology (partial realization)

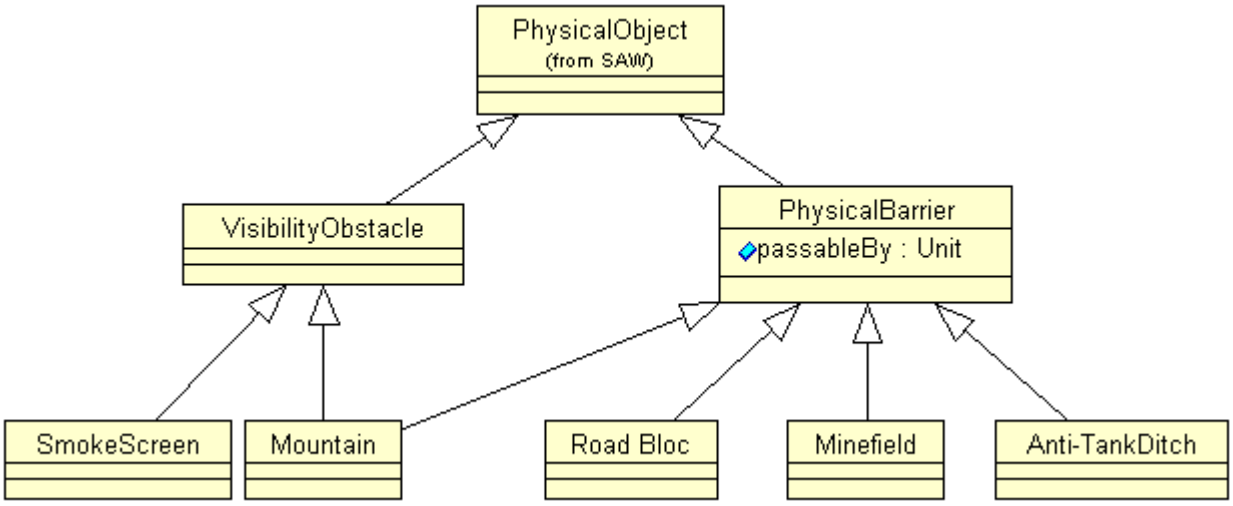

Figure 6 Battlefield Obstacle Ontology (partial)

6. SCENARIO

The Battlefield scenario used in our example consists of two simple snap-shots of events describing the initial interaction between two opposing tank platoons (see Figure 7). In the first snap-shot we observe a collection of three stationary blue

tanks (tank1, tank2, and tank3), an observation post and a minefield. In the next snap-shot (Figure 7), two red tanks (tank4 and tank5) appear approaching from the west and the three blue tanks begin advancing towards them.

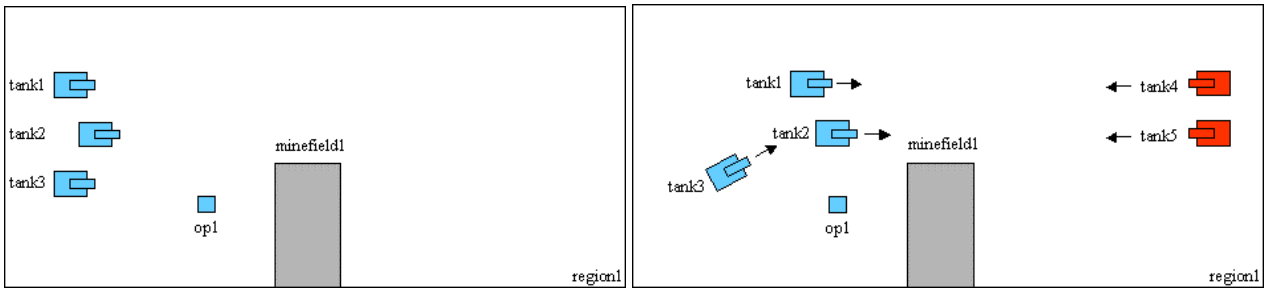

Figure 7 Battlefield Scenario Snap-Shots

To simulate the stream of EventNotices that would define the scenario we constructed two spreadsheets, one for each snap-shot, in which each row was a specific *EventNotice* about the *Attributes* of a particular *PhysicalObject* observed by a specific source at a specific time. These were saved as comma-separated-values (CSV) files and then converted into DAML instance annotations using Perl. To formalize these events in Specware we needed to translate them into statements representing axioms about the state of the situation at the respective times. This was done by converting the DAML annotations into MetaSlang statements with XSLT. Here is an example of a MetaSlang axiom for the effect caused by an *EventNotice* reported by *sensor1* at *12:50* concerning the *position* attribute of *tank1* with certainly of *0.9:*

```
axiom a1 is av(situation1,sensor1,Tank1,Position,1200)=PositionCons(makeVector(20,40,90))
```

The certainty factor here is encoded as part of the vector constructed to define the object's position.

With everything formally represented in Specware – including the goal, the relevant aspects of the SAW and Battlefield ontologies (*i.e.,* the classes, relations and rules relevant to the goal) and the relevant instance annotations from the scenario – we can now use SNARK to derive the higher-order relations that are evident in the scenario at particular points in time.

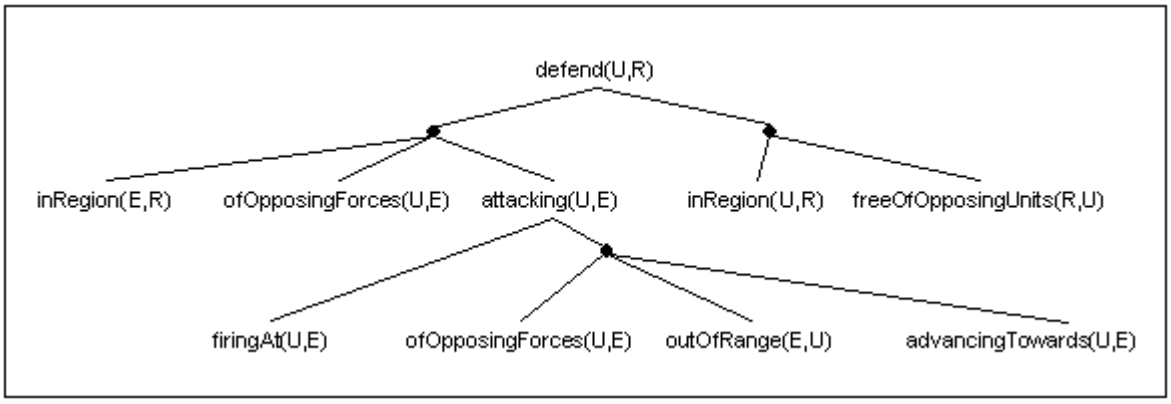

Figure 8 Partial Derivation Tree for Goal defend(U,R)

Figure 8 shows a partial derivation tree taken from the domain theory for the goal defend(U,R). If we consider the situation at the time of the first snap-shot and pose the conjecture *defend(tank1,region1)* the theorem prover will respond that it is in fact a theorem. The derivation of this result is achieved by a traversal of the right hand branch of the derivation tree. As a consequence two additional higher-order relations would have been derived: inRegion(tank1,

region1) and freeOfOpposingUnits(region1,tank1). If we pose the same conjecture at the time of the second snap-shot, it would again be confirmed as a theorem, but this time the left hand branch of the tree would be taken resulting in the derivation of several more relations including advancingTowards(tank1,tank5) or advancingTowards(tank1,tank6) and all of the additional relations occurring in rules required to satisfy these relations (not shown in the tree but present in the complete domain theory).

7. ISSUES AND OPEN QUESTIONS

Our proposed methodology provides a way to formally derive higher-order relations in a situation, as illustrated in the example in the previous section. There are a number of questions that emerge as we consider how this methodology would work in a fully-functional production system. There is the obvious question of complexity which our approach helps mitigate by focusing only on events and relations relevant to the goal. But does it go far enough to permit the creation of a system that could operate on non-trivial problems with real-time or near real-time performance? That is an open question which we intend to explore both empirically and theoretically in future work, although we have some preliminary thoughts on the matter. Our current approach employs backward-chaining to prove theorems which may not be the best approach giving the fact we need to be dealing with a continuous stream of incoming events. Since any given event might trigger the generation of a new relation derivation, or the destruction of an existing one, it probably makes more sense to base a practical implementation on a forward driven reasoning process. This is in fact our intention for the next phase of this work.

Our simple example side-stepped some of the more challenging details specified in our SAW ontology. For one thing, we did not talk about the use of dynamic systems to model changing *PropertyValues* that can change values themselves without the occurrence of a new *EventNotice*; such a capability is needed, for example, to predict the continually changing position of a moving object at some time after its last measurement. Even modeling simplistic dynamics (*e.g.,* a body in motion continues in the same direction at the same speed until informed otherwise by a new event) becomes a non-trivial task for a large ensemble of objects. We are encouraged, however, by the fact that this sort of thing is being done quite well in modern computer games, although not likely in a formal manner. Things gets worse when *PropertyValue* uncertainty is included, which, to be realistic, requires models for how certainty decays and these might need to be different for different types of attributes and relations.

Aside from performance concerns there are usability issues to consider. Assuming we can derive all (or a good percentage of) the relevant higher-order relations, how do we decide which of the derived relations might be of interest to the user? Taking the example from the previous section, we might contend that advancingTowards(tank1,tank5) is something a commander in the field would want to know about immediately, but would the details concerning why this is true – such as facing(tank1, tank5) – be of much interest? It would seem that the details of a derivation are not as important as the higher order relations they imply. On the other hand, there's a big difference between a tank defending a region by simply being present in the region (in the absence of enemies) and that tank defending the region as a result of it actually attacking the enemy (as implicated by the rules in Table 2). In this case some of the lower level details (*i.e.,* that tank1 is attacking something) are important. Some insight might come from focusing on things that change. For example, if the tank was defending the region at time t_1 and it was still defending it at time t_2 but it is now doing so because it began attacking an enemy, then clearly attacking the enemy is what is important and not the fact that it is still defending the region. This is a form of deviation detection whereby what is interesting is defined relative to what is expected or has happened in the past.[12] Another approach, which could be combined with the previous idea of deviation detection, would be to present to the user only the highest level relation of the new derivations and provide a facility for the user to ask for more information, in others words implement a *Why?* button.

Another user issue to be concerned with is that of the knowledge engineer who needs to construct domain ontologies. UML drawing programs are adequate for designing relatively simple ontologies of classes and relations but they are sorely lacking in other areas. What we need are ontology development environments that permit the construction and maintenance of ontologies, complete with the ability to 1) construct and easily navigate through large class diagrams, 2) import elements from other, external ontologies, 3) develop rules associated with and based on ontological classes, preferable in some easy-to-use, visual manner, and 4) generate code/mark-up in multiple formats (DAML/OWL, Rule-ML, Slang, etc). There are several efforts under way to tackle parts of this problem (the OntoWeb Consortium has

compiled a recent survey[13] of many of these), however, little has been done to include rules in ontology construction and maintenance tools; the fact that rules are not yet a part of the more popular ontology languages likely explains this omission.

To implement our example we needed to simulate the output of idealized Level 1 sensors. These simulations exploited the DAML ontology for the domain. We are not aware of any Level 1 sensors in existence today that have any notion of DAML (or any other ontology language) or are able to download ontologies that they can then use to represent the information they provide to upstream systems. We are confident, however, that techniques for delivering sensory information in DAML will exist in the not too distant future, as this exists as one of the objectives of the DARPA Information Exploitation Office[14] and is supported by the DAML Experiment[15].

8. RELATED WORK

Although situation awareness is very important in both military and commercial domains, research on information handling has been focusing mainly on Level 1 processing. Recently, however, developments in information retrieval, knowledge representation and automatic reasoning have paved the way for the development of systems for Level 2 processing. While the past data and information fusion conferences were almost entirely devoted to Level 1, the last Information Fusion Conferences in Montreal, Canada and in Annapolis, MD had a number of papers devoted to Situation Awareness. The starting point is the definition of SAW provided by Endsley[5]. Lambert[16] deals with philosophical issues of SAW and with the human-in-the-loop problem. He also presented an ontology-based formal approach to SAW, although he is pursuing the difficult undertaking of developing a universal ontology (in the paper he refers to an ontology of space and time). Another center for this kind of research is the Defense Research Establishment Valcartier in Canada (cf.[17,18]). Their approach is significantly different from the concept presented by our work. The authors focus on the process, i.e., on SA, rather than on SAW itself. This makes the approach less flexible because one needs to define all the procedural aspects upfront. Our approach, on the other hand, is declarative. We specify potential relations in an ontology and then leave the system design issues to the developers. A system then will be able to operate with any ontology, as long as it is represented in a given representation language (*e.g.,* DAML/OWL). Additionally, the advantage of our approach is that we provide a formalization of information fusion applicable to Level 2. This has not been proposed by anybody else.

9. CONCLUSION

The proof-of-concept we partially described in this paper demonstrates the applicability of our SAW methodology to the formal derivation of higher-order relations present in evolving situations. In military parlance this work demonstrates information fusion at level 2 of the data fusion hierarchy, also known as situation assessment. Because of its inherent complexity, this problem often has been viewed as intractable, which in the general case it is. A common technique to "solve" classes of intractable problems is to use heuristics or rules-of-thumb. Our approach uses the heuristic that the only things of importance in a situation are those things that are relevant to a given goal. With this heuristic we developed a formal methodology that is able to filter out a large portion of the search space used to discover relevant relations among objects in a situation. We discussed how we implemented our methodology using a SAW ontology which we formalized into specifications in Specware in order to do formal reasoning in SNARK. We then demonstrated how this would work in practice using a simple battlefield scenario. Future work in this area will focus on fully automating the systematic processes of our approach, developing intelligent tools to assist in the processes requiring human participation and working towards a practical, near-real-time implementation of an operational system.

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] L. von Mises, *Human Action: A Treatise on Economics*, originally published in 1949, Fox & Wilkes, 1997.
[2] M. Bunge, *Treatise on basic philosophy. III: Ontology: The furniture of the world*, Reidel, Dordrecht, 1977.
[3] J. Barwise, "Scenes and other situations", *J. Philosophy* 77**,** 369-397, 1981.
[4] J. Barwise, *The Situation In Logic*, CSLI Lecture Notes 17, 1989.
[5] M. Endsley and D. Garland, *Situation Awareness, Analysis and Measurement,* Lawrence Erlbaum Associates, Publishers, Mahway, New Jersey, 2000.
[6] 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.
[7] M. M. Kokar, J. A. Tomasik, and J. Weyman "Data vs. decision fusion in the category theory framework", In Proceedings of FUSION 2001 - 4th International Conference on Information Fusion, Vol .1, pages TuA3-15 - TuA3-20, 2001.
[8] M. M. Kokar and Z. Korona, "A formal approach to the design of feature-based multi-sensor recognition systems", International Journal of Information Fusion, 2 (2):77-89, 2001.
[9] H. Gao, M. M. Kokar, and J. Weyman, "An approach to automation of fusion using Specware", In Proceedings of the Second International Conference on Information Fusion,Vol.1, pages 109-116, 1999.
[10] K. Baclawski, M. Kokar, J. Letkowski, C. Matheus and M. Malczewski, "Formalization of Situation Awareness", In Proceedings of Eleventh OOPSLA Workshop on Behavioral Semantics, pp. 1-15, November, 2002.
[11] "Duet User's Guide", ATT Government Solutions Inc., 2002.
[12] G. Piatetsky-Shapiro, and C. Matheus, "The Interestingness of deviations", In Proceedings of AAAI Workshop on Knowledge Discovery in Databases, Seattle, WA, July 1994.
[13] OntoWeb Consortium, "OntoWeb Deliverable 1.3: A survey on ontology tools", May 2002.
[14] R. P. Wishner, "The Information Fusion Challenge", Presentation at Fusion'2002, 5-th International Conference on Information Fusion, 2002.
[15] J. Flynn and M. Dean, "DARPA Agent Markup Language (DAML) Demonstration and Experiment Plan", BBN Technologies, Version 0.8, CDRL A011, June 2002.
[16] D.A.Lambert, "Situations for situation awareness", In Proceedings of Fusion'2001, 4-th International Conference on Information Fusion, 2001.
[17] J.Roy, "From data fusion to situation awareness", In Proceedings of Fusion'2001, 4-th International Conference on Information Fusion ,2001.
[18] 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, 2001.