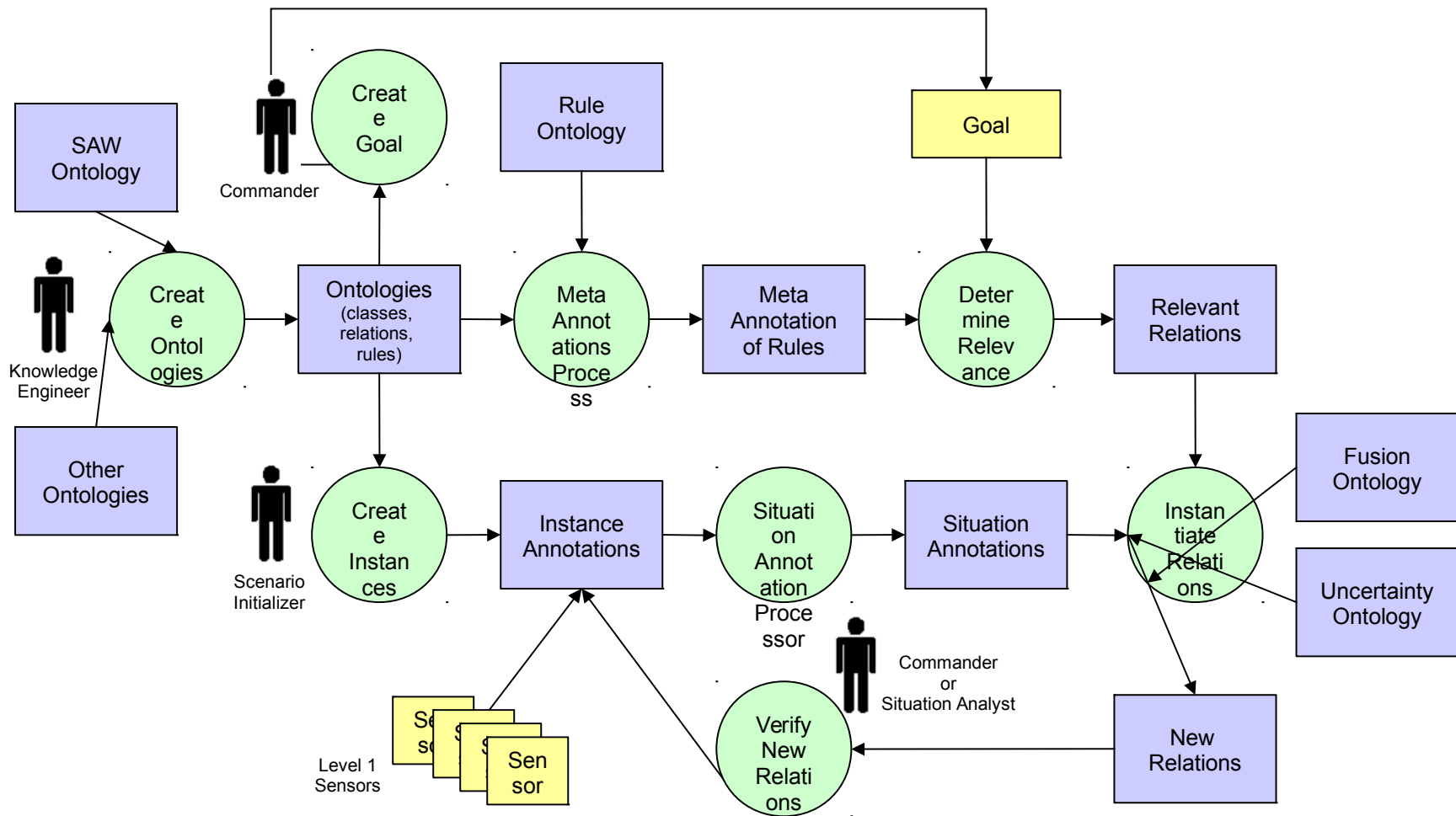


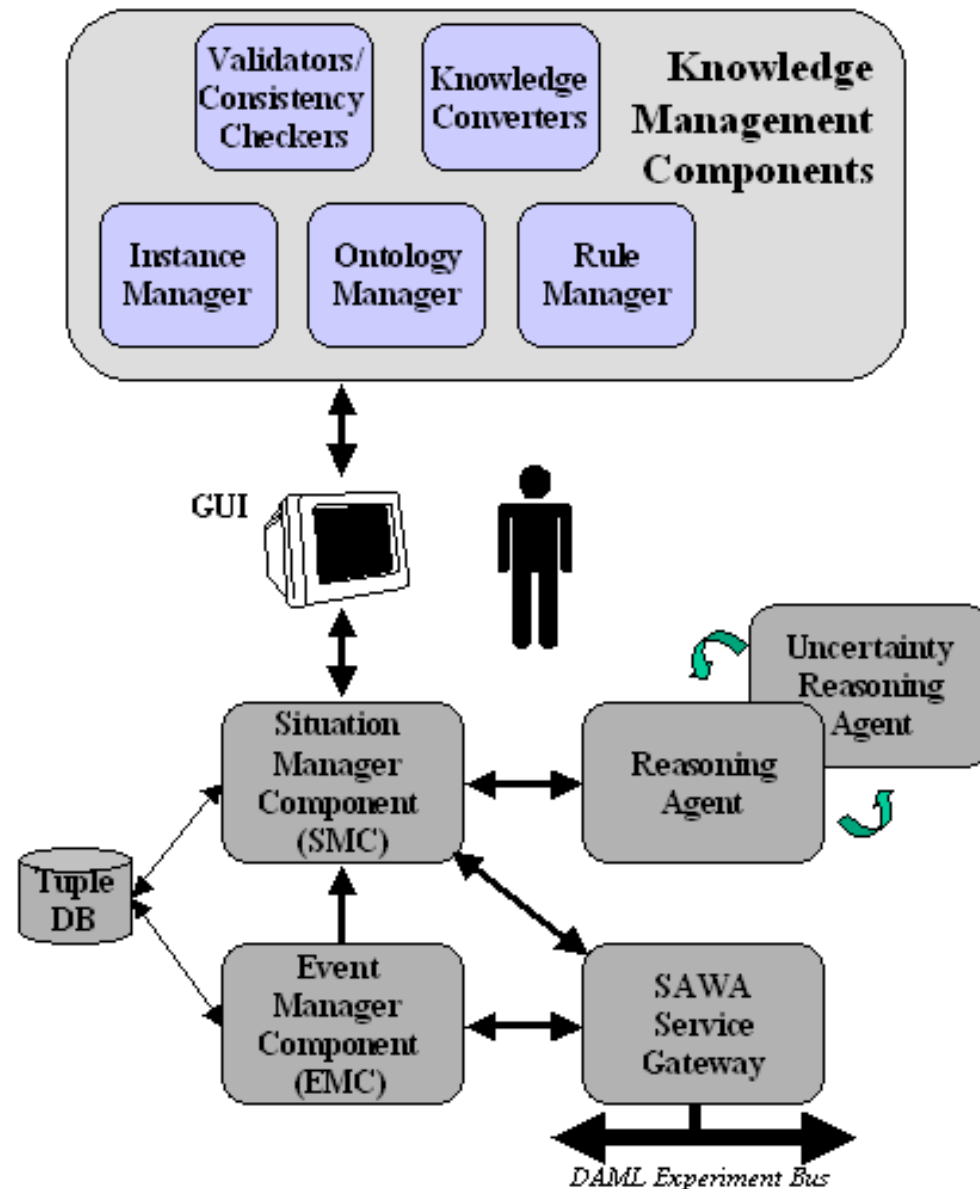
# SAWA: An Assistant for Higher-Level Fusion and Situation Awareness

Christopher J. Matheus, Mieczyslaw M. Kokar,  
Kenneth Baclawski, Jerzy A. Letkowski,  
Catherine Call, Michael Hinman, John Salerno,  
Douglas Boulware

# SAW Process



# SAW Assistant (SAWA)



# Supply Logistics Scenario

- Scenario for supplying units using ground transports via roads that may not be under friendly control
- Configuration files control types and quantities of resources, transports, suppliers and consumers
- Generates events based on our SAW Core, Supply Logistics and Event Ontologies

# OWL: Web Ontology Language

- W3C's ontology language for the Semantic Web
- Mainly intended to provide means for describing web content in a form amiable to automated reasoning
- Used to construct OWL ontologies that define domain specific classes and properties along with the inherent constraints among them
- OWL ontologies are then used to describe specific instances or situations in the given domain
- Built on top of RDF and XML
- Three flavors: Full, DL and Lite

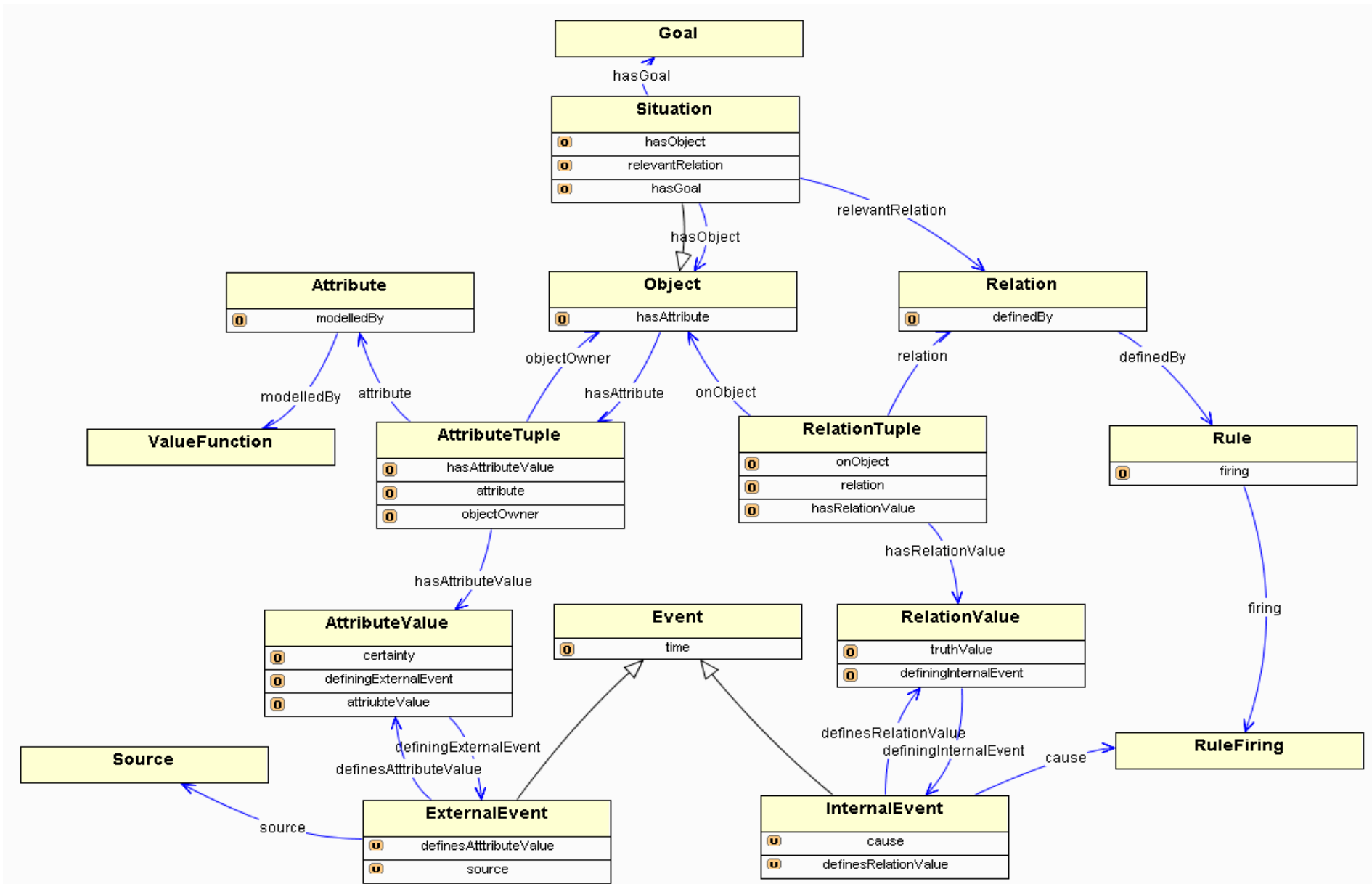
# SWRL

- W3C's Semantic Web Rule Language
- Extends representational power of OWL by adding implication in the form of Horn Clauses (i.e., a form of if-then rules)
- Leverages the descriptive capabilities of OWL DL
- Leverages the rule and variable syntax of RuleML

# SWRL Pros and Cons

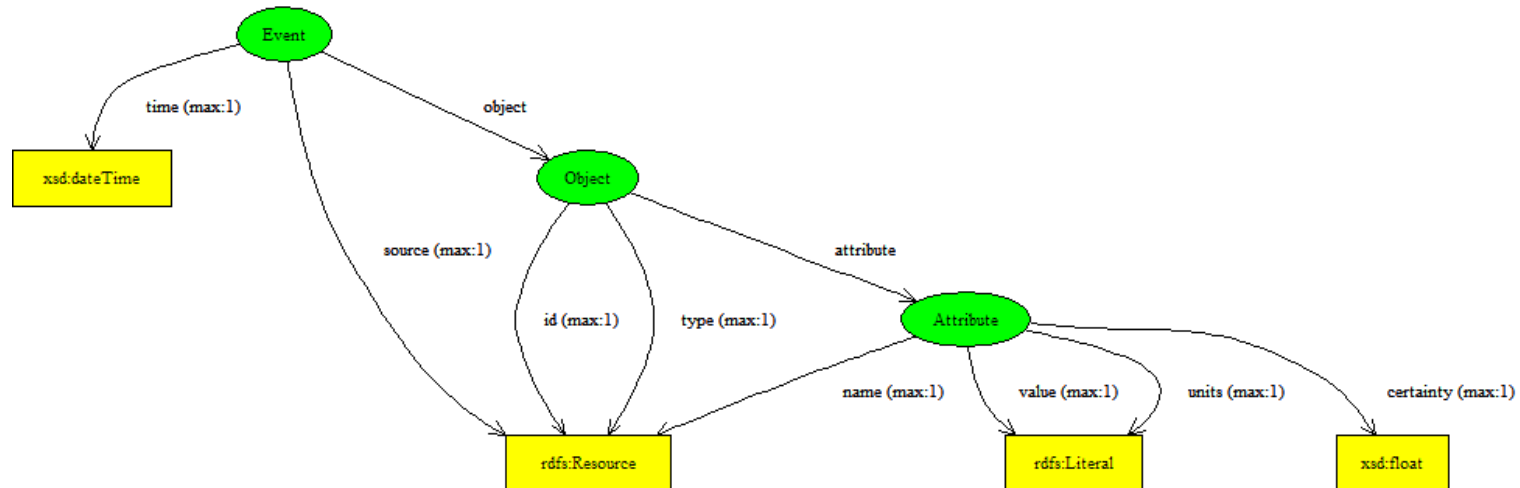
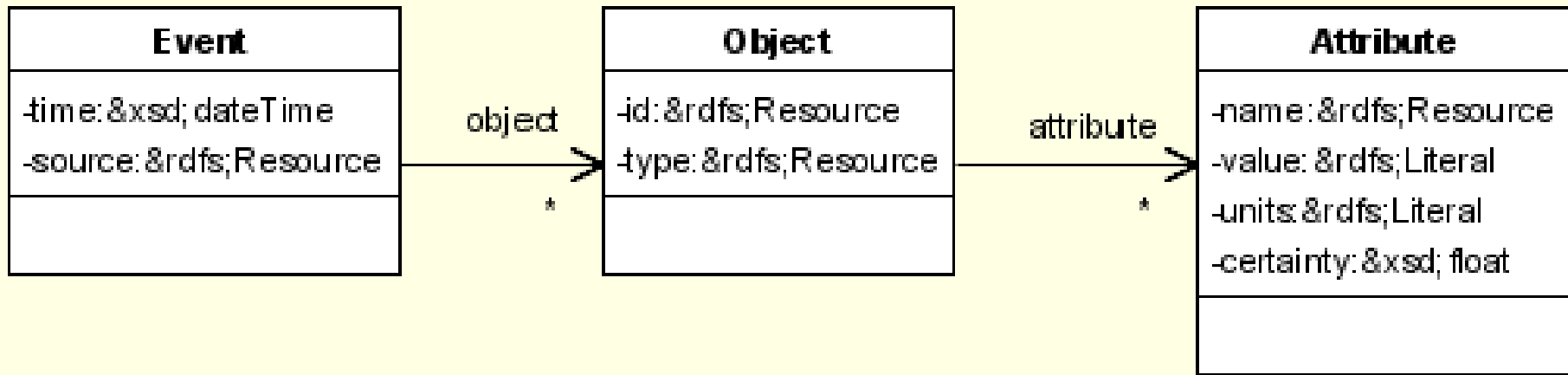
- Pros:
  - Formal Foundation
  - W3C Effort
  - Based on RuleML
  - Can connect to OWL Ontologies
- Cons:
  - Limited to Binary Relations (makes higher-order relations difficult to represent)
  - Verbose/complex syntax
  - No Existential Quantification in rule heads (makes higher-order relations impossible to infer – we thus are ignoring this constraint with the expectation it will be removed)
  - Still evolving

# SAW Core Ontology

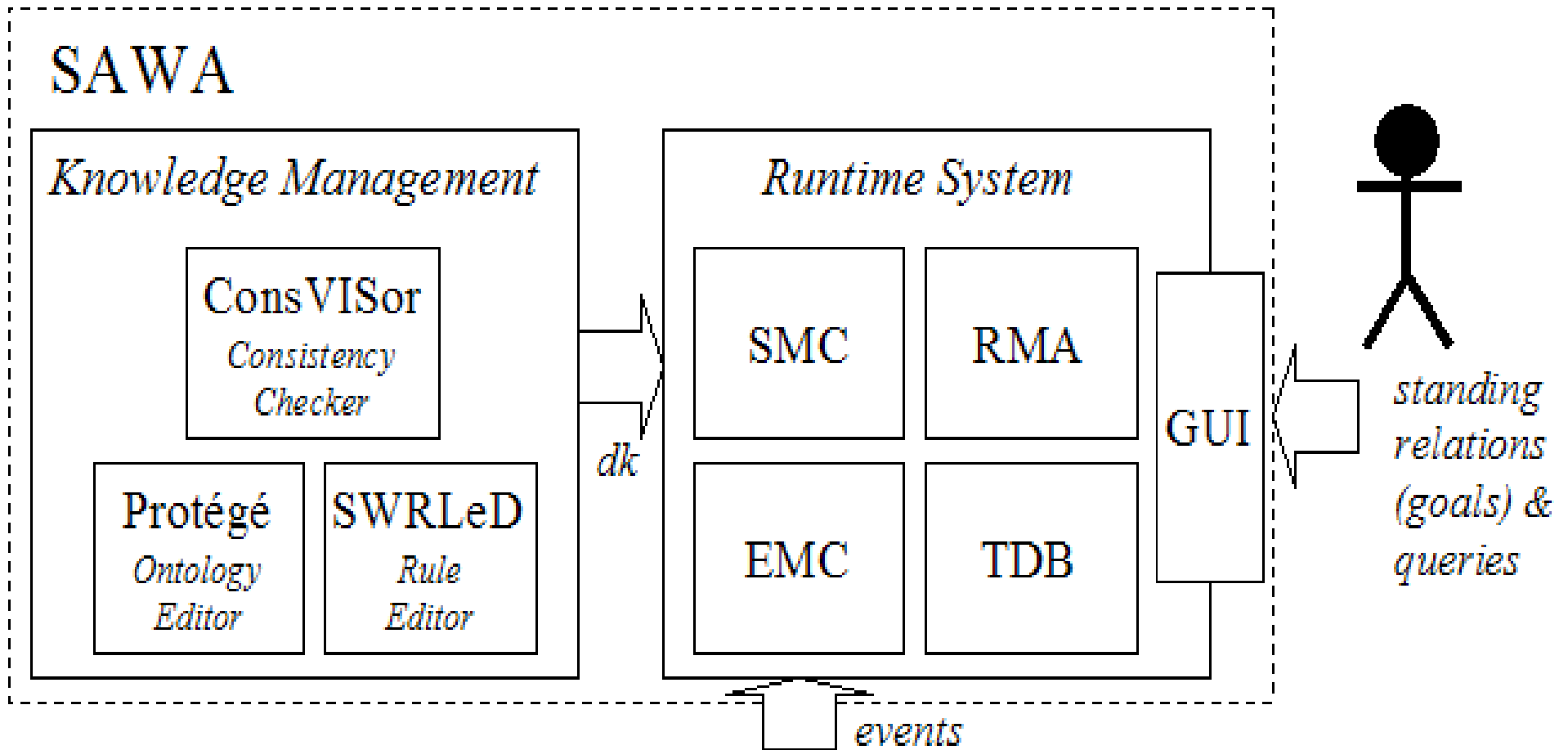




# Event Ontology



# SAWA Architecture



# ConsVISor Consistency Checker

ConsVISor Output - Microsoft Internet Explorer

File Edit View Favorites Tools Help defax

## ConsVISor Report

A service of [Versatile Information Systems, Inc.](#)

For <http://vistology.com/ont/tests/military3.owl>

### Consistency Verification Summary

<b>Test Outcome:</b>	The result of the consistency check of <a href="http://vistology.com/ont/tests/military3.owl">http://vistology.com/ont/tests/military3.owl</a> at level full is: inconsistent.
----------------------	--

[Messages](#)

Info	Warn	Error	Fatal
<a href="#">CardinalityConstraint</a>	No warning messages!	<a href="#">DisjointnessFailure</a> <a href="#">DisjointnessFailure</a>	No fatal error messages!

### Consistency Verification Detail

Resource Locations Are Shown in Brackets[row.col]

<b>Symptom:</b>	<a href="#">CardinalityConstraint</a>
<b>Message:</b>	A cardinality constraint with cardinality 1 was not satisfied.
<b>Axiom Violated:</b>	<a href="#">owl_cardinality</a>
<b>Severity:</b>	info

**Statements:**

Attribute	Subject	Predicate	Object
<i>sym:property</i>	b:_anon004 [a:22]	owl:onProperty	b:containedIn [a:51]
<i>sym:restriction</i>	b:Unit [a:19]	rdfs:subClassOf	b:_anon004 [a:22]
<i>sym:constraint</i>	b:_anon004 [a:22]	owl:cardinality	1
<i>sym:instance</i>	b:Easy2 [a:71]	rdf:type	b:Unit [a:19]
<i>sym:unsatisfied</i>	2	sym:equal	1

# RuleVISor Rule Editor

- Graphical SWRL Editor
- Support for
  - all RuleML capabilities (everything in SWRL from ruleml: namespace)
  - all new SWRL elements (from swrlx: namespace, e.g., swrlx:builtin)
- Does not support arbitrary embedded OWL constructs
  - OWL Ontologies are required to be external
- Ontologies used as basis for rule building blocks

# RuleVISor GUI

The screenshot displays the RuleVISor SWRL Rule Editor interface. The title bar reads "RuleVISor SWRL Rule Editor" and the status bar indicates "Editing Ruleset: C:\Chris\VIS\SAWA\Scenario\hasSupplyLineRules.swrlx".

The interface is divided into several sections:

- Left Panel:** Contains a file explorer showing the project structure and an "Ontology Tree" listing namespaces and classes like `hsl:SupplyStation`, `hsl:EnemyForce`, `hsl:Road`, `hsl:Unit`, `hsl:Region`, `hsl:_anon005`, `hsl:Force`, and `hsl:FriendlyForce`.
- Top Panel:** Shows tabs for different rules: `has Supply Line`, `isSuppliable`, `isSuppliable2` (selected), `underFriendlyControl`, `isPassable`, and `hasSupplyStation`.
- Rule Editor:** The main workspace for editing the selected rule. It includes:
  - Head:** A table defining the rule's head. The rule name is `isSuppliable2`. It contains two rows: one for the `add` function with arguments `?depthPlus1` and `?depth` (both of type `xsd:int`), and another for the `datavaluedProperty` `hsl:isSuppliable` with subject `?region1` and object `?depthPlus1` (domain `hsl:Region`, range `xsd:int`).
  - Body:** A table defining the rule's body. It contains four rows: two for `hsl:connects` (subject `?road`, domain `hsl:Road`, range `hsl:Region`), one for `notEqual` (term1 `on1`, term2 `on2`, both of type `xsd:anyURI`), and one for `hsl:isPassable` (subject `?road`, object `true`, type `xsd:boolean`).

# Supply Logistics Rule Set

```
<rule rlab="has Supply Line">
  <body>
    <hsl:inRegion      sub="?unit"      data="?region"/>
    <hsl:isSuppliable sub="?region"    data="true"/>
  </body>
  <head>
    <hsl:hasSupplyLine sub="?unit"      data="true"/>
  </head>
</rule>

<rule rlab="isSuppliable">
  <body>
    <hsl:hasSupplyStation sub="?region"  data="true"/>
    <hsl:underFriendlyControl sub="?region" data="true"/>
  </body>
  <head>
    <hsl:isSuppliable sub="?region"    data="true"/>
  </head>
</rule>

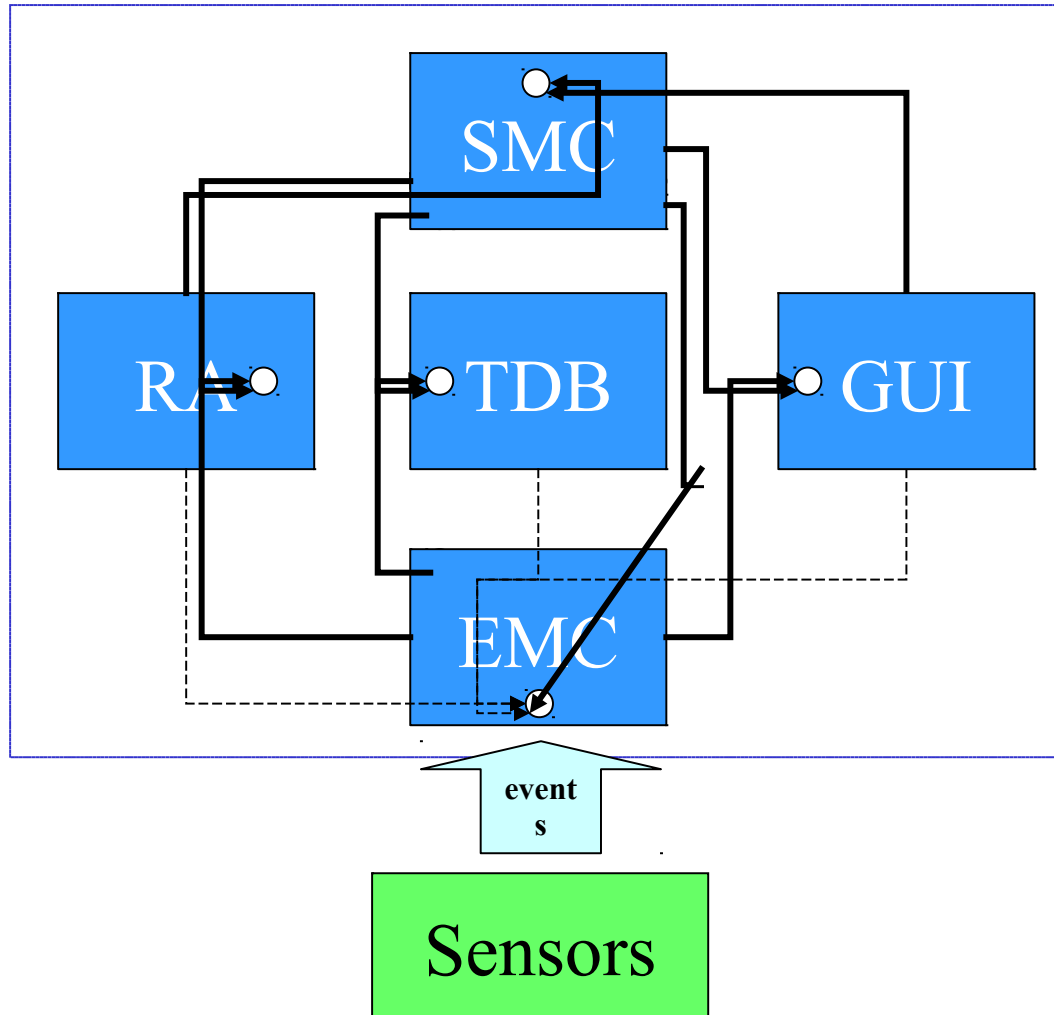
<rule rlab="isSuppliable2">
  <body>
    <hsl:connects      sub="?road"      data="?region1"/>
    <hsl:connects      sub="?road"      data="?region2"/>
    <swrlb:notEqual
      arg1="?region1"
      arg2="?region2"/>
    <hsl:isPassable   sub="?road"      data="true"/>
    <hsl:isSuppliable sub="?region2"  data="true"/>
  </body>
  <head>
    <hsl:isSuppliable sub="?region1"  data="true"/>
  </head>
</rule>
```

```
<rule rlab="underFriendlyControl">
  <body>
    <hsl:inRegion      sub="?unit"      data="?region"/>
    <hsl:memberOf      sub="?unit"      data="?force"/>
    <hsl:FriendlyForce ind="?force"/>
  </body>
  <head>
    <hsl:underFriendlyControl sub="?region" data="true"/>
  </head>
</rule>

<rule rlab="isPassable">
  <body>
    <hsl:connects      sub="?road"      data="?regionA"/>
    <hsl:connects      sub="?road"      data="?regionB"/>
    <swrlb:notEqual
      arg1="?regionA"
      arg2="?regionB"/>
    <hsl:underFriendlyControl sub="?regionA" data="?force1"/>
    <hsl:underFriendlyControl sub="?regionB" data="?force2"/>
  </body>
  <head>
    <hsl:isPassable   sub="?road"      data="true"/>
  </head>
</rule>

<rule rlab="hasSupplyStation">
  <body>
    <hsl:inRegion      sub="?X"          data="?region"/>
    <hsl:SupplyStation ind="?X"/>
  </body>
  <head>
    <hsl:hasSupplyStation sub="?region"  data="true"/>
  </head>
</rule>
```

# SAWA Runtime



# Triple Data Base

- Stores RDF/OWL triples
  - E.g., (predicate subject object)
- Built on Jess (Java Expert System Shell based on CLIPS)
- Infers implicit triples from events and OWL axioms
- Detects inconsistencies
- Tracks performance metrics of inference engine
- Supports OWL-QL (OWL Query Language) formerly known as DQL



# Query Capabilities

- Full support of OWL Query Language – DARPA sponsored effort
- Permits Queries over patterns in triples
  - e.g., (consumes ?user “food”) (type ?user “company”)
  - Results returned as variable bindings
- “What If” Query capability
  - assumptions posited and then retracted after query returns
- Writing queries and interpreting results can be challenging
- Prompted move to implement simple GUI

# Query Interface

- Simplifies query construction
- Initial version based on static templates with fill-in slots
- Demo
- Extensions:
  - constraints between slot values enforced by GUI
  - automatic generation of candidate templates
  - free-form query wizard

# Query GUI Screenshot

Monitor for Situation ID=S1, Goal=SupplyAsSoonAsPossible

New Goal: SupplyAsSoonAsPossible Query ? Start Pause Resume Stop

### Current Event

- Event(E1). Source(supplier-pt). Date(2004-03-24). Time(23:35:27.92)
  - Object(supplier-pt). Type(Supplier)
    - Name(Type). Value(Supplier). Units(class). Certainty(1.0)
    - Name(PositionX). Value(-90.0). Units(point). Certainty(1.0)
    - Name(PositionY). Value(20.0). Units(point). Certainty(1.0)
    - Name(VelocityX). Value(0.0). Units(mph). Certainty(1.0)
    - Name(VelocityY). Value(0.0). Units(mph). Certainty(1.0)
    - Name(SupplyOnHand). Value(Food-PT-S1). Units(id). Certainty(1.0)
    - Name(Produces). Value(ProductionInfo-Food-supplier-pt). Units(id). Certainty(1.0)

### Object Table

Object	Attribute	Value	Units	Certainty	Event	Date	Time
supplier-pt	Type	Supplier			E1	2004-03-24	23:35:27.92
	Type	Supplier	class	1.0			
	PositionX	-90.0	point	1.0			
	PositionY	20.0	point	1.0			
	VelocityX	0.0	mph	1.0			
	VelocityY	0.0	mph	1.0			
	SupplyOnHand	Food-PT-S1	id	1.0			
Food-PT-S1	Type	Supply			E1	2004-03-24	23:35:27.92
	Type	Supply	class	1.0			
ProductionInfo-Food-supplier-pt	Supply	Food	SupplyType	1.0			
	Amount	300.0	pounds	1.0			
	Type	SupplyRate	class	1.0			
	Rate	1.0	seconds	1.0			
supplier-pt	Type	Supplier			E1	2004-03-24	23:35:27.92
	Type	Supplier	class	1.0			
	PositionX	-90.0	point	1.0			
	PositionY	20.0	point	1.0			
	VelocityX	0.0	mph	1.0			
	VelocityY	0.0	mph	1.0			
	SupplyOnHand	Food-PT-S1	id	1.0			

### Visual View

SAWA Query

Query

In Supply

Which suppliers supply  and how much is in supply?

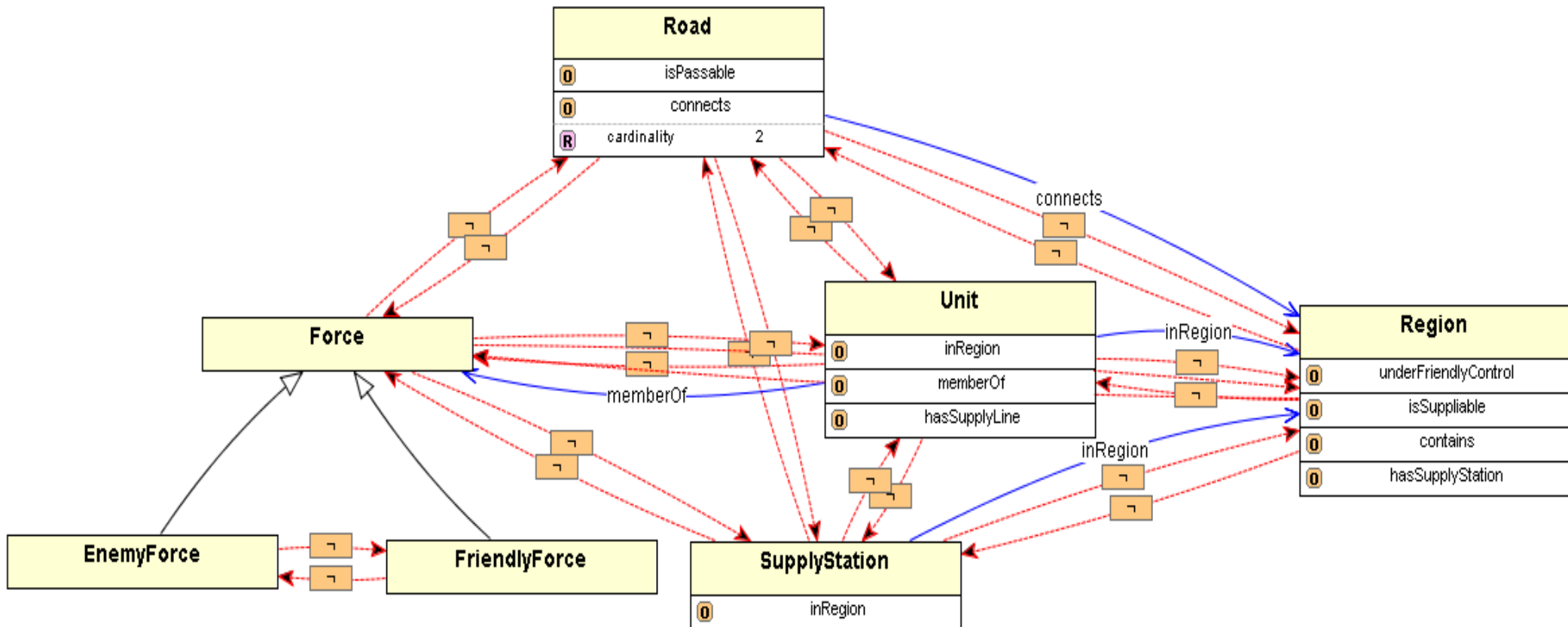
Run

Supplier	Supply	Amount
Sup1	Food	300
Sup2	Food	200
Sup3	Food	250

### Relevant Relations

Relation(X,Y)	Value	Certainty	Object X	Object Y	Event

# Supply Logistics Ontology



# SAWA Runtime GUI

Situation Monitor ID=S0, Goal=hasSupplyLine

File Help

Standing Relation: hasSupplyLine

Supply Lines of Units  Passable Routes  Suppliable Regions

Next  Stop

**Current Event**

ID E3 Time 2004-06-08 18:30:19.06 Source S0:sensor1

Inference Depth: 4.1 Rule Firing: 107 Assertion Rate: 18

**Relevant Relation Diagram**

**Situation Object Map**

**Relevant Relations Table**

Relation(X,Y)	Probability	Object X	Object Y	Value	Units	Certainty
hasSupplyLine	1.0	B5	B5	True		true
hasSupplyLine	1.0	B7	B7	True		true
hasSupplyLine	1.0	B8	B8	True		true
hasSupplyLine	1.0	B9	B9	True		true
isPassable	1.0	Route3	Route3	True		true
isPassable	1.0	Route4	Route4	True		true
isPassable	1.0	Route8	Route8	True		true
isSuppliable	1.0	RegionA	RegionA	True		true

**Object Information Dialog**

About: <http://www.vistology.com/ont/2004/supply/hasSupplyLineScenario#B8>

Object ID	B8
Object Type	FriendlyForce
Last Event	E2
Time	2004-06-08T17:30:19.06Z

Attribute	Value	Units	Certainty
PositionX	180.0	point	1.0
PositionY	210.0	point	1.0
VelocityX	0.0	mph	1.0
VelocityY	0.0	mph	1.0

**Situation Object Table**

Object	Type	Attribute	Value	Units	Certainty	Event	Date	Time
RegionG	Type		Region			E1	2004-06-08	16:30:19.06
RegionH	Type		Region			E1	2004-06-08	16:30:19.06
RegionI	Type		Region			E1	2004-06-08	16:30:19.06
RegionJ	Type		Region			E1	2004-06-08	16:30:19.06
RegionK	Type		Region			E1	2004-06-08	16:30:19.06
RegionL	Type		Region			E1	2004-06-08	16:30:19.06
RegionM	Type		Region			E1	2004-06-08	16:30:19.06
Route1	Type		Road			E1	2004-06-08	16:30:19.06
Route2	Type		Road			E1	2004-06-08	16:30:19.06

# Conclusion

- SAWA is a general purpose assistant for situation awareness:
  - monitors the evolution of relevant higher-order relations within a situation.
  - supports formal reasoning techniques for level-2 fusion.
  - based on the Semantic Web languages OWL and SWRL.
  - performs relevance reasoning.
- The domain ontology and rules are constructed and checked using an ontology editor, rule editor and consistency checker.
- At runtime events are processed to determine relevance and to infer higher-order relations.
- As higher-order relations are detected they are passed to the GUI, which displays them in both tabular and graphical forms.
- The query capability allows for both ordinary and “what if” queries.