16th International Command and Control Research and Technology Symposium
"Collective C2 in Multinational Civil-Military Operations"
Québec City, Canada, June 21–23, 2011

The Multi-Intelligence Tools Suite – Supporting Research and Development in Information and Knowledge Exploitation

Jean Roy
Defence R&D Canada – Valcartier
2459 Pie-XI Blvd. North,
Quebec (Quebec), Canada, G3J 1X5
jean.roy@drdc-rddc.gc.ca

Dr. Alain Auger
Defence R&D Canada – Valcartier
2459 Pie-XI Blvd. North,
Quebec (Quebec), Canada, G3J 1X5
alain.auger@drdc-rddc.gc.ca

Paper #118
Track 4: Information and Knowledge Exploitation

Point of Contact
Jean Roy
Defence Scientist
Group Leader – Sensemaking Group
Intelligence and Information Section
Defence R&D Canada − Valcartier
2459 Pie-XI Blvd. North
Quebec (Quebec)
Canada, G3J 1X5
Email: jean.roy@drdc-rddc.gc.ca
DWAN: JEAN.ROY@forces.gc.ca
Telephone: 1 (418) 844-4000 / Ext. 4379
Facsimile: 1 (418) 844-4538
www.valcartier.drdc-rddc.gc.ca

The Multi-Intelligence Tools Suite – Supporting Research and Development in Information and Knowledge Exploitation

Jean Roy
Defence R&D Canada – Valcartier
2459 Pie-XI Blvd. North,
Quebec (Quebec), Canada, G3J 1X5
jean.roy@drdc-rddc.gc.ca

Dr. Alain Auger
Defence R&D Canada – Valcartier
2459 Pie-XI Blvd. North,
Quebec (Quebec), Canada, G3J 1X5
alain.auger@drdc-rddc.gc.ca

Abstract

While fulfilling its research mandate, the Intelligence and Information Section at DRDC Valcartier is constantly developing computer-based tools to support the analysts involved in intelligence activities. These tools are developed under different research projects, for various customers in diverse domains (e.g., improvised explosive devices and maritime situational awareness), to address specific aspects (e.g., the semantic analysis of unstructured documents, the use of automated reasoning to infer anomalous behaviours, etc.). For a large portion, they are built on knowledge-based systems technologies. However, only providing stovepipe tools is not optimal; some integration is also required to create a synergy among them and facilitate the work of the analysts. The Multi-Intelligence Tools Suite (MITS) has thus been created as a federation of innovative, composable and interoperable intelligence related tools, which are integrated and interleaved into an overall, continuous process flow relevant to the intelligence community. At the software system level, the backbone of the MITS is an integration platform built on open source Web services technologies, following service-oriented architecture (SOA) design principles. The paper first reviews the main characteristics of the MITS. Then it discusses the central notions of domain knowledge and situational facts, describes the ingestion in the MITS of structured and unstructured data and information, briefly describes the main modules of the MITS, provides an exploitation example highlighting some of its powerful and innovative capabilities, and introduces the SOA platform and human-computer interaction components that constitute the MITS.

1. Introduction

Intelligence refers to a special kind of knowledge necessary to accomplish a mission, i.e., the kind of strategic knowledge that reveals critical threats and opportunities that may jeopardize or assure mission accomplishment [Waltz, 2003]. Intelligence is knowledge and foreknowledge of the world around us, the prelude to decision and action. In this rapid changing world, the expectations required of those in the intelligence discipline are high. The consumers of intelligence all expect accurate and timely information about their areas of interest and threats to their security. The process that delivers strategic and operational intelligence products is generally depicted in cyclic form [Waltz, 2003], with distinct constituents for obtaining, assembling and evaluating information, converting it into intelligence, and disseminating it [McIntyre et al, 2003]. The processing phase of the intelligence cycle involves the collation, evaluation, analysis, integration and assessment of the gathered information. The organized information base is processed using estimation and inferential (reasoning) techniques that combine all-source data in an attempt to answer the requestor's questions [Waltz, 2003]. The data is analyzed (broken into components and studied) and solutions are synthesized (constructed from the accumulating evidence).

A key aspect of the research activities conducted by the Intelligence and Information (I&I) Section at Defence R&D Canada – Valcartier (DRDC Valcartier) has to do with the development of computer-based tools to support the operators/analysts involved in the activities of the intelligence cycle. A number of individual tools have been developed under different research projects, for various customers in diverse domains (e.g., the domain of countering improvised explosive devices (IEDs), the maritime situation awareness domain etc.). Each tool addresses a specific aspect, such as the semantic analysis and automated annotation of unstructured documents, or the use of rule-based automated reasoning for the generation of alerts to draw the attention of the operators/analysts on some anomalous behaviour of some actors in a monitored situation. New tools are also continuously being developed.

From an operational perspective, only providing individual, specific, stovepipe tools is often not the most optimal way to proceed. Some tool integration is also required to create a synergy among them and to facilitate the work of the intelligence operators/analysts. In this regard, the I&I Section at DRDC Valcartier has created the Multi-Intelligence Tools Suite (MITS). The MITS is a federation of innovative, composable and interoperable intelligence related tools, which are integrated and interleaved into an overall, continuous process flow relevant to the intelligence community. It is worth mentioning that given the intrinsic nature of the intelligence domain activities, and that of the related research activities of the I&I Section in knowledge and information management and exploitation (KIME), most of the tools in the MITS involve knowledge-based system and semantic Web technologies.

This paper discusses the MITS from different perspectives, presenting it as a key component supporting research and development in information and knowledge exploitation in the intelligence domain. The paper is organized as follows. Section 2 first reviews the main characteristics of the MITS. Then Section 3 discusses the central notions of domain

knowledge and situational facts. The ingestion in the MITS of structured and unstructured data and information is discussed in Section 4. The main modules of the MITS are briefly described in Section 5, while Section 6 provides a detailed exploitation example highlighting some of its powerful and innovative capabilities. Section 7 introduces the SOA platform and human-computer interaction components that together constitutes the MITS, and Section 8 finally provides some concluding remarks.

2. Key Characteristics of the MITS

This section reviews the high-level characteristics of the MITS, which were initially identified and specified even before the first implementation of the MITS. To a certain degree, these characteristics could be considered as high-level requirements for the MITS.

2.1 Multi-Int Analysis

The primary focus of the research mandate of the I&I Section at DRDC Valcartier is "Multi-Int Analysis", which can be defined as "maximising the use of multiple sources of information to make better intelligence products and outcomes". As a result, the numerous computer-based support tools developed through the various R&D activities of the I&I Section are, more than often, of a "multi-int" nature from the start. An important requirement for the MITS is thus to enable the ingestion of intelligence products generated by sources covering the wide variety of the "INT" spectrum (IMINT, HUMINT, COMINT, ELINT, SIGINT, OSINT, MASINT, etc.). Here, the notion of "intelligence products" is important. There is no requirement for the MITS to be able to ingest raw data and to perform "low-level" processing, such as image processing or signal processing for example. It is rather expected that some pre-processing will be achieved somewhere else to generate intelligence products prior to their importation into the MITS. This being said, an important aspect of the MITS has to do with the ingestion and processing of unstructured text documents, some of which can be considered as "raw data".

2.2 Tool Suite: One-Stop Shop for the Tools

The MITS constitutes a "one-stop shop" for the numerous computer-based support tools developed by the I&I Section. From the intelligence analyst point of view, it is advantageous to easily have access to all available tools in a centralized location as they become required for the analysis problem of the day. Otherwise, it could quickly become cumbersome for an analyst to have to localize the individual tools first, potentially on different systems, and then to invoke them and transfer data and information between them.

2.3 Tool Synergy

Another main goal of organizing and providing the tools as a suite is to generate some synergy among them, for the benefit of the analysts. Initially, each tool is typically studied, design and developed in isolation, to address some specific aspects of the intelligence process cycle that have been clearly identified as important by the intelligence community. In general, it is thus optimized to tackle these aspects and, when everything goes as expected, it is in itself considered a valuable asset for the analysts. However, it is also highly valuable to enable the mutually advantageous conjunction or compatibility of the distinct intelligence analysis elements offered by these various tools. The targeted « cliché » is that the output of the integrated suite is greater than that of the sum of the standalone stovepipe tools taken individually.

2.4 Ease of Exploitation

A major requirement for the MITS is certainly to make the exploitation of all these available tools easy for the analysts. In this regard, an important benefit of having the tools organized as an integrated suite is that it facilitates the transfer of data, information and knowledge « products » from one tool to the other. The concept of the MITS enables the seamless pipelining of the individual tools to achieve more global capabilities. Also related to the ease of exploitation is that the potential contribution of any given tool to information and knowledge exploitation in the intelligence domain must be easy to understand and must be made known to the analysts. The tools must be easy to access and configure. The organization of the tools into meaningful categories must help the analysts to quickly find the tool(s) that they really need, at the moment they really need it.

2.5 Uniformity of Exploitation

Providing some uniformity in the exploitation of the tools is another key aspect to achieving efficiency in the support to the analysts, and it is a targeted characteristic of integrated environments such as the MITS. The organization in the MITS of the situational and application domain knowledge manipulated by the analysts contributes to the homogeneity of exploitation. The ontologies used to model the different application domains of interest and the situational facts generated and managed during the analysis process have been centralized and made accessible to all of the tools of the suite that manipulate them. This is also reflected in the graphical user interface (GUI) of the MITS for which various icons and human-computer interaction (HCI) mechanisms have been carefully specified and used to standardize the main items of the knowledge framework that are common to most of the tools of the suite. In addition to simplifying the configuration and execution of the system, the GUI facilitates the exchange of data, information and knowledge between the tools.

2.6 An R&D Integration Environment

Right from the start, it was decided that the MITS would be an R&D environment, not an operational environment. That is, the MITS exists to support and integrate research and development in the intelligence domain, not to be deployed as is in some operational environment. The MITS is used to explore innovative, advanced concepts, and to evaluate and demonstrate the potential value of these concepts. If some "gold nuggets" are found along the way using the MITS during the research activities, then they become candidates for transition into the field using other platforms targeted for operational use. It may happen that some components of the MITS could be transitioned and reused almost as is in the operational environment, but this should not be the main goal pursued with the MITS itself.

2.6.1 Unconstrained Environment

Very often when developing a system for the operational environment, technological constraints arise from various perspectives, and they are usually quite different than those dictated by research objectives. The reality of the operational world is such that some technology may be imposed to the developers because of legacy systems already in the field (e.g., it would be too costly to remove a component or even just modify its implementation), because of safety or security considerations, because of political reasons (e.g., the government has issued a politic that favour a given sector of the industry), or because of many others quite legitimate reasons. This reality can quickly become a serious limitation on the space of scientific and technological possibilities that must be considered when conducting research activities. From a research perspective, there are often technologies worth exploring that are, at the time of their exploration, incompatible with the operational environment. In this regard, the MITS is seen and developed as an environment where the scientists and developers have no constraint on the technologies that can be used and explored.

2.6.2 Reusability and Incremental Development

Incremental S&T knowledge acquisition, discovery and creation are inherent to the research process. A spiral development approach is often adopted in the conduct of research projects and to deliver research programs. It is thus very important that scientists and developers work on a platform that supports the incremental development of components. And this is directly linked to the notion of reusability, as « *not having to start from scratch every time* » is also quite important while developing new ideas in a framework with severe budget limitations. Hence, reusability and incremental development were two additional key elements justifying the MITS environment.

2.6.3 Integrated Validation

Exhaustive testing and evaluation of the intelligence support tools developed is typically achieved in multiple steps. Certainly, any new component must first be tested as a standalone piece, in some « unitary test », to check if it works as expected. However, it is also essential to test and validate any new tool in the context of the other components of the overall system to cover some critical aspects that may be missed in unitary testing.

First of all, testing a tool in isolation may require the creation of some specific stimulation component to feed this tool (e.g., the creation of an appropriate dataset). In turn, the level of realism of this stimulation component must be sufficient to support a credible evaluation. Unfortunately, the level of realism required to really evaluate a tool may be hard to achieve (e.g., too costly, sometime even more costly than that of the new tool being developed) if testing is only considered as a separate, standalone activity. In an integrated environment like the MITS, the high-quality components already developed and validated by « the others » may be used to provide realistic stimulation for the new component being introduced and tested. Aspects that are considered « peripheral » to a given tool are potentially covered by the other components of the system because they are considered as « central » aspects from the perspective of these other components. Hence, one has to care less about such peripheral aspects, which are then taken in charge by others. Ultimately, all system components become high quality elements. Overall, this contributes to reducing the requirements on high-fidelity simulations that otherwise would have been necessary to test a particular tool.

Secondly, it is also necessary to evaluate the behaviour of the overall set of components when they interact with each others to achieve the objectives of the analyst. In particular, it is important to focus at the impact of the new component being introduced. One must not only check that the new tool delivers the products of interest, but also that it doesn't generate any undesirable side effects once integrated.

2.7 Knowledge-Based Systems Technologies

A knowledge-based system (KBS) is a computer system that represents and uses knowledge to carry out a task [Stefik, 1995]. An expert system is an intelligent computer program that uses knowledge and inference procedures to solve problems that are difficult enough to require significant human expertise for their solution. As the applications for the technology have broadened, the more general term knowledge-based system has become preferred by some people over expert system because it focuses attention on the knowledge that the systems carry, rather than on the question of whether or not such knowledge constitutes expertise.

For a large portion, the intelligence processing tools of the MITS have been built on KBS technologies. The selection of these technologies has been motivated by a number of their intrinsic characteristics, the main one being that processing is separated from the problem-solving knowledge in knowledge-based systems. This characteristic allows: 1) to represent knowledge in a more natural fashion, 2) the focus to be on capturing and organizing problem-solving knowledge, 3) changes to be made to the knowledge base without side effects on program code, 4) the same control and interface software to be used in a variety of systems, in different domains, and 5) to experiment with alternative control software for the same knowledge base. As a result of the attributes mentioned above, the processing components of a KBS are typically generic (i.e., the processing is intrinsically « agnostic »; it's the a priori knowledge of a particular domain that makes the processing specific), developed "only once" (or more precisely, the exact same components can be used/reused in different application domains without any modifications being required), and developed by "others" (i.e., they are developed, tested, debugged, etc. by others and then made available from open sources or commercially).

The benefits and issues related to the use of knowledge-based systems technologies in the context of situation analysis support systems (SASSs) have already been discussed at length in [Roy, 2006], [Roy, 2007-A] and [Roy, 2007-B]. The key related aspects of knowledge representation concepts, paradigms and techniques, and reasoning processes, methods and systems, and knowledge and ontological engineering techniques for use in developing knowledge-based SASSs were further investigated in [Roy, Auger, 2008-A], [Roy, Auger, 2008-B] and [Roy, Auger, 2008-C] respectively.

In the context of the MITS, the use of KBS technologies has allowed the scientists at DRDC Valcartier to first develop a single, unique system, and then to exploit it under different research projects, for various customers in diverse domains (e.g., improvised explosive devices and maritime situational awareness).

3. The MITS as a Knowledge-Based System

As discussed avove, given the intrinsic nature of the intelligence domain activities, and that of the related research activities of the I&I Section in knowledge and information management and exploitation (KIME), most of the tools in the MITS involve knowledge-based system and semantic Web technologies.

3.1 Knowledge Representation

One cannot put the world in a computer, so all reasoning mechanisms must operate on representations of facts, rather than on the facts themselves [Russell, Norvig, 1995]. The object of knowledge representation (KR) is to express knowledge in computer-tractable form, such that it can be exploited [Russell, Norvig, 1995]. KR and reasoning is the area of artificial intelligence (AI) concerned with how knowledge can be represented symbolically and manipulated in an automated way by reasoning programs [Brachman, Levesque, 2004]. KR research studies the problem of finding a language in which to encode the knowledge so that the machine can use it [Ginsberg, 1993]. It should support the tasks of acquiring and retrieving knowledge, as well as subsequent reasoning [Turban, Aronson, 1998].

One may have the impression that a knowledge engineer must find a single best representation and stick with it. However, it is not necessary to select and use only one representation in a knowledge system [Stefik, 1995]. Actually, no single knowledge representation method is ideally suited by itself for all tasks [Turban, Aronson, 1998]. An important alternative is the use of multiple representations. A variety of knowledge representation paradigms, schemes and techniques have been devised in the AI community over the years [Roy, 2006]. These includes lists and outlines, decision tables, decision trees, state and problem spaces, production rules, object-attribute-value triples, semantic networks, schemata, frames, scripts, logics, etc.

To support the activities of the analyst in the intelligence domain, the MITS allows for the manipulation of a number of knowledge representation building blocks: ontologies, facts, inference rules, text-based templates, etc. Some of these are further discussed next.

3.1.1 Ontologies

During the last decade or so, increasing attention has been focused on ontologies and ontological engineering [Gómez-Pérez et al, 2004]. The word ontology was taken from Philosophy, where it broadly means a systematic explanation of being. Ontologies have become relevant for the knowledge engineering community, and many definitions of the word have been proposed by a variety of authors. Among these, the definition proposed by [Struder et al, 1998] based on the work of [Gruber, 1993] seems appropriate for the development of knowledge-based SASSs: "An ontology is a formal, explicit specification of a shared conceptualization." Within the MITS, an ontology is a formal representation of a set of concepts within a domain, and of the relationships between those concepts. It is used to define a domain and to reason about the properties of this domain.

3.1.2 Facts and Atom Definitions

In natural languages, a conventional unit of expression is the sentence [Stefik, 1995]. In knowledge representation, the term *sentence* is used as a technical term, which is related to the sentences of English and other natural languages, but is not

identical [Russell, Norvig, 1995]. Sentences in logic are intended to tell us things about the world [Ginsberg, 1993]. In this context, each individual representation of facts about the world is called a sentence, and basic sentences are called atoms [Russell, Norvig, 1995]. Based on these simple concepts, the notions of a « fact » and an « atom definition » have been adopted as key knowledge representation paradigms for the MITS.

By definition [Merriam-Webster, 2003], a fact is something that has actual existence or an actual occurrence; it is a piece of information presented as having objective reality. Within the MITS, a fact is a pragmatic truth, a statement that can, at least in theory, be checked and confirmed. The left-hand side of Fig. 1 provides an example of a simple fact, expressed in a natural language (English), in a formal language that is more suitable for manipulation in a computer program, and also using the concept of an atom, as is used in the MITS.

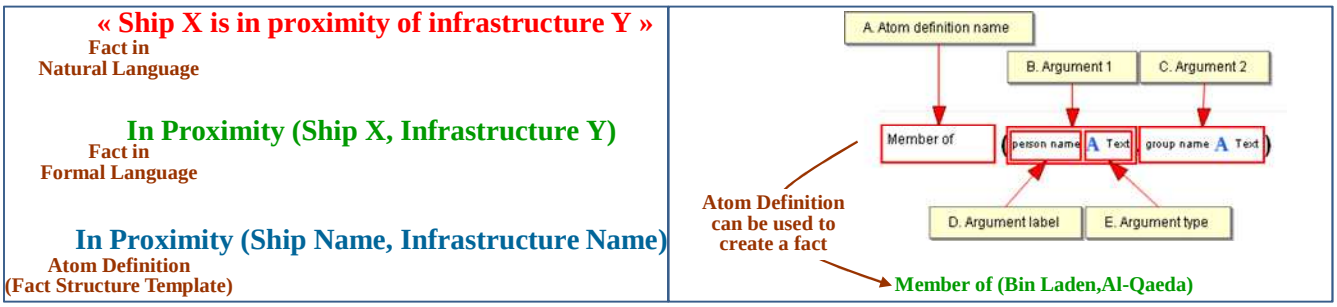

Figure 1. A fact (expressed in different languages) and an atom definition [DMR, 2010]

Within the MITS, as shown on the right-hand side of Fig. 1, an atom definition specifies the structure of a fact (i.e., a fact is a structured assertion based on the atom definition). The atom definition is actually the base structure of any fact used in MITS. It's a formal template defining which pieces are required to form a fact. It is defined by a name and a list of arguments with a precise type and order. For example, if one needs to manipulate facts in the MITS asserting the membership of a person to a group, one must first create an atom definition that will support this requirement, like the one shown in Fig. 1.

Each atom definition has a name (A) to represent the fact, and argument definitions (B and C) that represent the kind of pieces that needs to be defined to form the fact. For example, a membership fact (Fig. 1) can't exist if one only has a person or if one only has a group. Both pieces are required to build a membership fact. The atom definition also defines the precise order of the pieces. Each argument has a label (D) and a type (E). The argument label is used to distinguish the different pieces required, and the argument type is used to restrict the value that can be set for this argument. There are two argument types available for argument definition: 1) instance type argument (refers to an instance type of a knowledge domain's ontology, or any child entity), and 2) literal argument (refers to any other non-ontology entity value; however, for this type of argument definition, one must specify the value type among the available literal types: text, Boolean (true or false), integer, decimal number, date, date range, area, and distance). Atom definitions are used in inference rules, text-based templates for fact extraction, and many other components of the MITS that manipulate facts.

3.1.3 Built-In Definitions

A built-in definition has the same structure as an atom definition. It is composed of the same elements (argument definitions and so on). However, the nature and purpose of a built-in definition is different. First, built-in definitions can only be used in the premises of inference rules to perform argument value validations. No fact instance can be generated directly from a built-in definition. In contrast with atom definitions used in inference rule premises and that are validated by trying to match them against existing facts in the fact knowledge base, built-in definitions are validated more like invoking a function. Furthermore, since a function is required in order to perform the validation of the built-in definition, the system is pre-initialized with a predefined set of built-in definitions. They cannot be edited and deleted, and it is not possible for the user to add a new built-in definition.

Built-in definitions are evaluated at runtime and do not require to have a fact defined for each of their argument to be able to evaluate their equation. This avoids creating tons of useless facts to do simple comparisons. The current built-in definitions are: after/before (takes two date arguments and verifies if the first date is chronologically after/before the second date), collect (takes one atom element, and two integer constraints arguments. It counts the number of facts respecting the atom definition. The integer arguments are optional; they make it possible to define constraints for the count value to establish the premise as valid. One can set 0, 1 or 2 integer arguments. Operators are also considered. For example if one sets 1 integer argument having a value of 5 with an operator type "Greater than", then the premise will only be valid if there are more than 5 atoms with the specified definition. If one adds a second argument with a value of 10 with an operator type "Less Than", then the premise will only be valid if there are between 5 and 10 atoms with the specified definition), includeDate (takes one date range argument and one date argument and verifies if the date is contained in the date range), includeDateRange (takes two date range arguments and verifies if the second date range is contained in the first date range), inArea (takes two area

arguments and verifies if the second area is contained within the first area), overlap (takes two date range arguments and validates if the two dates ranges overlap).

3.1.4 Inference Rules

Inference rules are used by the automated reasoning engine to infer new facts. An inference rule defines which pattern of facts will generate new facts. Such a pattern of facts can be seen as domain specific knowledge (typically obtain from a domain expert) specifying which facts should be deduced based on the existence of other facts. For example, one may want to automatically create a fact « Driving infraction » referring to a person if one already has the facts « License expired (person name) » and « Currently driving (person name) ».

Each inference rule is composed of premises (a list of atom definitions and/or built-in definitions with constraints on arguments values) and conclusions (a list of atom definitions defining which facts to generate, with values to be set for each argument). Figure 2 shows the inference rule editor.

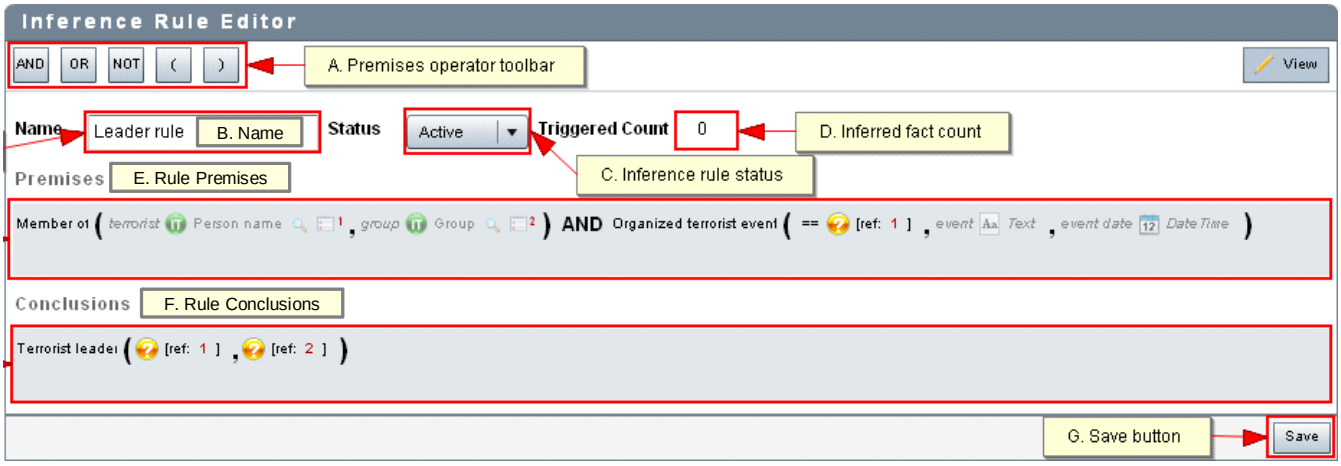

Figure 2. Inference rule editor [DMR, 2010]

One must type in a name for the inference rule (B), and set the inference rule status (C) as « Active » (to make the rule usable by the inference engine), or « Disabled » (to prevent the inference engine using this rule until the status is set back to «Active»). The triggered count field (D) contains the number of inferred facts directly created using the rule. This field is not editable; it is automatically incremented whenever a new inferred fact is created using this specific rule. One must set at least one premise in the rule premises list (E), which represents a list of fact conditions that need to be found in the fact knowledge base in order to trigger the rule and generate its conclusion. To perform this action, one must drag an atom definition (or a built-in definition) from the knowledge engineering toolbox and drop it in the premises list. A premise with empty arguments will be automatically created. For each argument of the created premise, one has three choices:

1. Leave the argument empty: This means that no restriction will be set on the value of this argument.
2. Define an argument value (instance, instance type, instance type pattern or literal value): This allows restricting the facts that the inference engine will try to match with the premise to only those who have the same argument value. Moreover, for a literal argument, one can select a comparison operator that will be used for the restriction.
3. Use a reference to another argument ( ) in the premises as a dynamic value constraint: This allows comparing facts argument values to further filter the premise facts. To do this, one must drag any compatible argument (argument of the same type, and in the same ontology sub-tree in the case of an ontology entity) from the premises that are preceding the target argument, and then drop it on the target argument. This will add a small reference number next to the dragged argument and an argument reference will be created on the target argument with the same reference number. This allows quickly viewing which argument is referenced. Moreover, when one moves the mouse cursor over the argument reference, the referenced argument is highlighted at the same time.

Built-in definitions can also be dragged in the premises list of a rule, which allow performing more advanced validations on the argument values. For example, validating if two literal arguments of type date range overlaps can be achieved by adding to the premises list the built-in definition « overlap ». One must define an operator between each inference rule premise by dragging the appropriate operator from the premises operator toolbar and dropping it in the premises list. Parenthesis can be added to force the evaluation order. This allows specifying a more complex validation pattern. The “NOT” operator can also be used, which either negate a built-in condition, or make sure no fact is matching a certain premise atom.

To complete the rule, one must add at least one conclusion representing the fact that will be generated if the premises restrictions are met. To perform this action, one must drag an atom definition from the knowledge engineering toolbar in the

rule conclusions list (F). Built-in definitions cannot be used in the conclusions list. For each conclusion added to the conclusions list, one must define the argument values, i.e., 1) predefine an argument value by dragging an instance, instance type, instance type pattern or literal argument from the argument value toolbar on a compatible argument of a conclusion; every inferred fact created using this rule will then have the same argument value, or 2) use a value from a premise argument by dragging an argument from the premises and dropping it on a compatible conclusion argument; the referenced value of the dragged argument will be automatically assigned to the inferred fact corresponding to the conclusion argument reference.

The analyst may organize the inference rules into one or multiple inference rules sets, which are aggregations of one or more inference rules that have a common purpose or domain of application.

3.1.5 Text-Based Templates for Automated Fact Extraction from Unstructured Text Documents

Text-based templates are used by the text processing module of the MITS to find precise series of words in unstructured text documents (Word, PDF, etc.) and to extract specific facts from them. They are composed of text-based elements that needs to be matched against text contents, and a list of conclusions defining which fact(s) to generate when the pattern is matched. Each conclusion also specifies which values will be set in the generated facts arguments. As an example, one could create a text-based template stating that whenever « a person name followed by any membership expression (*is member of*, *is part of*, etc.) followed by a group name » is found in a text document, the system must generate a fact based on the atom definition «*Member of*» having the person name found in the text as the first argument, and the group name found in the text as the second argument. Figure 3 shows a different example of a text-based template for fact extraction.

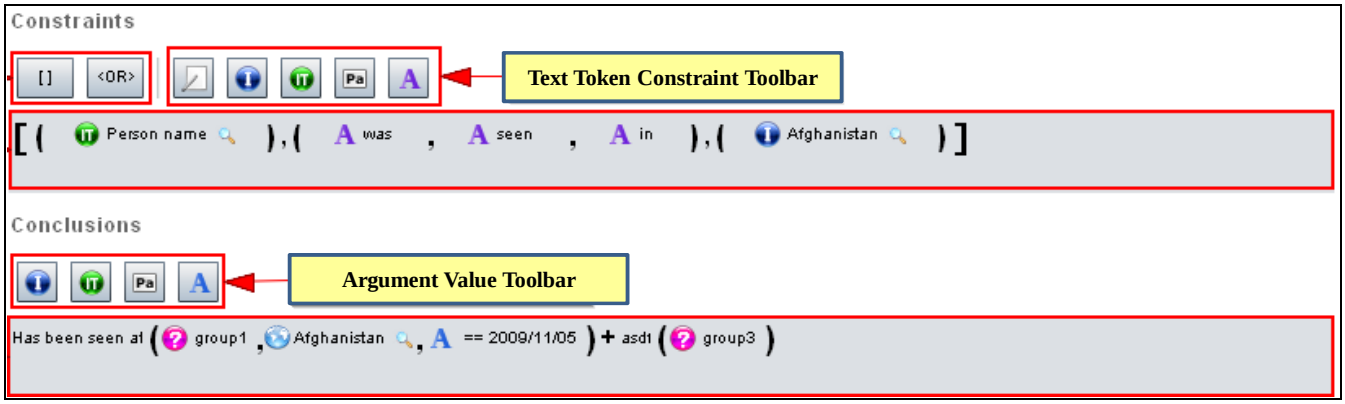

Figure 3. Example of a text-based template for fact extraction [DMR, 2010]

One must set the text-based pattern that has to be found in a text document in order to trigger the extraction and generate the fact(s) defined in the text-based template conclusions. This pattern is called the text-based template constraints. When a document is processed, the text processing module first extracts the text from the document and then splits this text into multiple parts composed of one or many words within the sentence. Each part is named a *text token*. The same word may be used in more than one token. The main types of aspects in which the text is split are: 1) ontology entity occurrence (representing any ontology entity textually found in the text, 2) lexical function of words (representing a lexical category function of a word or a group of word), and 3) word splitting (every word will also be split for string matching). The editor allows one to build complex pattern constraints by using groups and operators. After defining the appropriate text-based constraints, one must define the fact(s) that will be created when the constraints are detected in a document by dragging in the list of conclusions the atom definition(s) corresponding to the fact(s) to create. The analyst may organize the text-based templates into one or multiple template sets, which are aggregations of one or more text-based templates that have a common purpose or domain of application. Details regarding automated fact extraction from unstructured text documents can be found in [Auger, 2009].

3.2 Domain Knowledge (A Priori / Reference / « Static »)

The techniques being developed for information fusion and situation analysis are becoming increasingly more sophisticated, particularly through the incorporation of « intelligent » processes at a very high level of abstraction [Roy, 2006]. Until the mid-sixties, a major quest of artificial intelligence was to produce intelligent systems that relied little on domain knowledge and more on powerful methods of reasoning [Giarratano, Riley, 1998]. By the early 1970's, it had become apparent that domain knowledge was the key to building machine problem solvers that could function at the level of human experts. It is well established now that the first step in solving any problem is defining the problem area or domain to be solved [Giarratano, Riley, 1998].

Along this line of thoughts, a fundamental component of information fusion and situation analysis techniques and methods is a database (or databases) containing a priori domain knowledge that lists such things as expected objects, behaviours of objects, and relationships between objects [Boury-Brisset, 2001]. A situation analysis support system should analyze and

combine observations of the current situation within the context of this a priori domain knowledge. The expression “a priori knowledge” here actually includes static (or slowly changing) knowledge/information/data (KID) to support the (or required by) various information fusion and situation analysis processes. It refers to different aspects that can be used by situation analysis modules. [Boury-Brisset, 2001] discussed research activities to extend the traditional and conventional concept of a support database for information fusion and situation analysis to the one of a knowledge management and exploitation server product. She described the functional architecture of such a server that makes use of ontologies and heterogeneous knowledge sources, and also reviewed some engineering aspects for its construction.

3.2.1 Domain Environment Knowledge

In the context of the discussion above, a « *domain* » is a body of knowledge [Stefik, 1995]. In knowledge representation, a «*domain*» is a section of the world [Russell, Norvig, 1995]. Within the MITS, a knowledge domain is valid knowledge used to refer to an area of human endeavour, an autonomous computer activity, or another specialized discipline. It can be represented by one or a set of ontologies. Figure 4 shows some examples of domain knowledge ontologies currently available in the MITS.

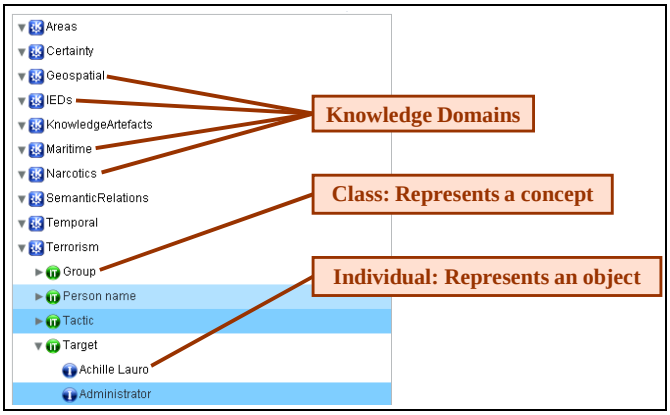

Figure 4. Knowledge domains ontologies in the MITS [DMR, 2010]

Many components of the MITS are related to these domain ontologies (atom definitions, text-based templates, inference rules, etc.). Instance types and instances are the MITS materialization of classes and individuals in the OWL formalism. Within the MITS, an instance type pattern is an entity representing multiple instances by defining a label text pattern.

3.2.2 Domain Expert Knowledge (Know-How)

Expertise is a specialized type of knowledge that is known only to a few [Giarratano, Riley, 1998]. It is not commonly found in public sources such as books and papers. Instead, expertise is the extensive, task-specific and implicit knowledge of the expert that is acquired from training, reading, and experience, and that must be extracted and made explicit so it can be encoded in an expert system. An expert is a person who has (or is recognized by peers as having) expertise in a certain area. One actually talks about degrees or levels of expertise. In general, the term expert connotes both specialization in narrow problem-solving areas or tasks and substantial competence [Stefik, 1995], [Turban, Aronson, 1998]. An expert can solve problems that most people cannot solve, or can solve them more efficiently (but not as cheaply) [Giarratano, Riley, 1998].

An expert's knowledge is specific to one problem domain, as opposed to knowledge about general problem-solving techniques. Expertise in one problem domain does not automatically carry over to another. Expert systems are generally designed to be experts in one problem domain. Actually, restricting the problem domain is typically necessary to produce useful solutions.

While conducting the research activities of the I&I Section at Valcartier, problem-solving know-how is first acquired from domain experts through interviews and other knowledge acquisition methods, and then represented (or encoded) within the MITS as *a priori/reference/« static »* ontologies, inference rules, text-based templates for fact extraction, etc.

3.2.3 Knowledge Base and Domain Knowledge « Cartridge »

A knowledge base (KB) is the organized repository for the collection of knowledge related to a domain and used for understanding, formulating, and solving problems in a knowledge-based system [Stefik, 1995]. The basic elements that are usually included in the KB are [Turban, Aronson, 1998]: 1) facts, such as the theory of the problem area, 2) special heuristics, rules and hints that direct the use of knowledge to solve specific problems in a particular domain, and 3) global strategies, which can be both heuristics and a part of the theory of the problem area. Once a KB is built, AI techniques are used to give the computer an inference capability based on the facts and relationships contained in the KB. That is, the KB contains data structures that can be manipulated by an inference system that uses search and pattern matching techniques on the KB to answer questions, draw conclusions, or otherwise perform an intelligent function. With a KB and the ability to draw

inferences from it, the computer can be put to practical use as a problem solver and/or decision assistant. By searching the KB for relevant facts and relationships, the computer can reach one or more alternative solutions to the given problem, in support to the analysts. The KB can be organized in several different configurations to facilitate fast inferencing (or reasoning) from the knowledge [Turban, Aronson, 1998].

Within the MITS, each knowledge domain is implemented using a set of knowledge domain descriptors. Actually, the concept of using « *domain knowledge cartridges* » has been developed and implemented in order to support various analysis needs. A knowledge cartridge contains all necessary resources to describe and formalize a knowledge domain for use in the MITS system: ontologies and taxonomies, local grammars (pattern matching rules), inference/alerting rules, text-based templates and fact generators, sources characterization. For example, the MITS supports improvised explosive devices (IEDs) threat assessment tasks by the automatic exploitation of an IED ontology, sets of IED atom definitions, facts extraction rules and IED threat assessment rules (inference rules).

An important conclusion is that ultimately, the technology components in the MITS can be used to investigate intelligence problems within very specific knowledge domains, or across multiple domains in problem spaces involving multiple knowledge domains.

3.3 Situation Knowledge Base (Dynamic Situation Model)

[Roy, 2001] has defined situation analysis (SA) as a process, the examination of a situation, its elements, and their relations, to provide and maintain a product, i.e., a state of situation awareness, for the analysts and/or decision makers. Clearly, operational trends in warfare and public security activities put the SA process under pressure. Information technology support is thus typically required to cope with the human limitations in such complex environments. This emphasizes the need for real-time, computer-based situation analysis support systems (SASSs) to bridge the gap between the cognitive demands inherent to the accomplishment of the SA process and the human limitations.

The main purpose of a SASS is to assemble a representation of aspects of interest in an environment. A SASS should thus incorporate and develop an internal situation model of the environment of interest. This situational model, that the SA process endeavours to keep up to date, captures not only the representation of the various elements of the situation, but also a representation of how they relate to create a meaningful synthesis supporting the comprehension of the situation [Roy, 2001]. There is one real world, and the situation model is an abstraction of it supporting situation awareness for the analyst/decision maker. As the idea of *awareness* has to do with *having knowledge of something* [Merriam-Webster, 2003], the situation model thus has to do with situation knowledge representation.

During the (supported) analysis of a situation, some model of this situation is constructed within the MITS by augmenting the a priori domain knowledge ontologies with instances observed or inferred to be part of the situation, and by establishing situational facts (through direct conversion from the observations, or through other means such as inference). Actually, situational facts and situational ontologies are interrelated as the situational facts are often statements made about some instances of the domain knowledge ontologies.

3.4 Knowledge Engineering Module and Knowledge Exploitation

Figure 5 illustrates how the various knowledge representation building blocks discussed above in this section are interrelated and exploited in the different modules of the MITS that are concerned with fact generation.

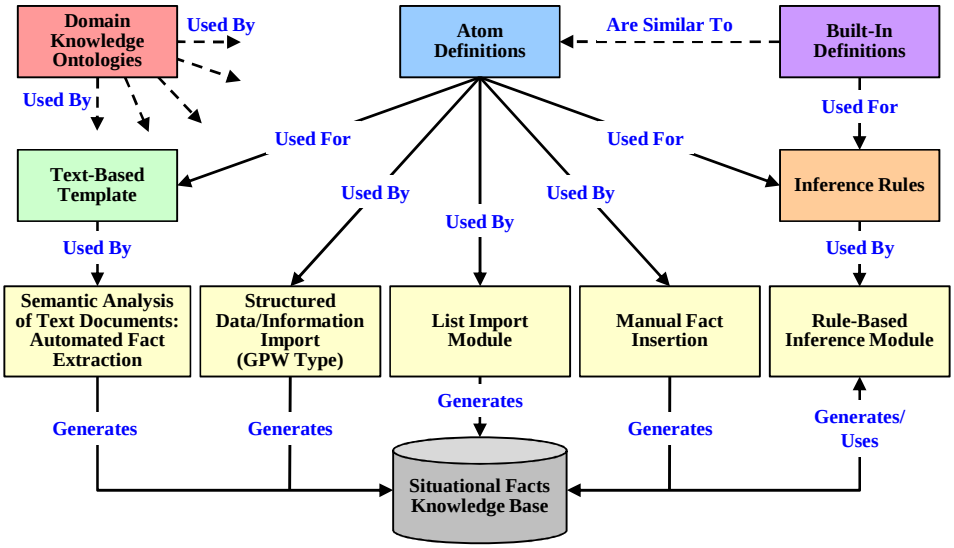

Figure 5. Knowledge representation building blocks and situational facts generation in the MITS

As shown on Fig. 5, facts are generated: 1) by the text processing module exploiting text-based templates and unstructured text documents, 2) by the rule-based inference engine exploiting inference rules and previously existing facts, 3) by the analyst through manual fact insertion, 4) through the conversion of input data and information from structured sources, and 5) through the importation of subjects from subject lists.

To make the MITS useful, knowledge must be acquired from experts and formally represented and stored in knowledge bases to be exploited by the various KIME tools. To support these knowledge acquisition and representation activities, a knowledge engineering module has been developed and implemented for the MITS. It provides a user-friendly graphical user interface (GUI) that make the definition and specification of atom definitions, inference rules, text-based templates, etc. easy for the knowledge engineer and/or the analyst.

Figure 6 shows an important component of the MITS knowledge engineering module, the Related Assets Viewer, which is used to display the assets that are related to a selected entity. This interface is divided in two sections: 1) related assets history (which displays the last items that have been consulted), and 2) related assets diagram (displaying which assets are related to the current entity). This interface allows performing the following main actions: visualize the entity metadata and its related assets, consult paged related assets results, visualize related assets of a previously consulted entity, and clear the related assets history. The related assets can be of the type fact, fact subscription, text-based template set, and/or inference rule set.

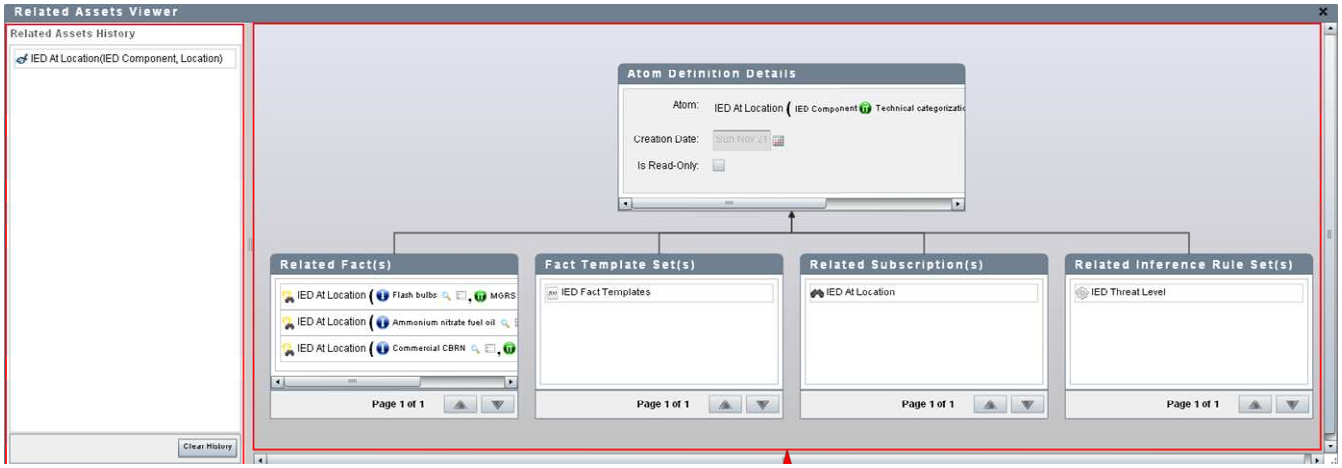

Figure 6. Atom definitions related assets viewer [DMR, 2010]

4. Ingesting Data into the MITS

Figure 7 illustrates how external data are ingested into the MITS. The system consumes structured and unstructured documents provided by a wide variety of data sources. Structured documents/sources include track data, databases, Excel worksheets, XML documents, etc. Since these documents are structured, it is possible to extract more precise information and metadata about them. One « source adapter/converter » per type of structured document/source must be developed. The documents are then processed by these adapters/converters so that their content and metadata can be inserted into the MITS knowledge base. Usually, the structured source processing components generates facts or modifies the knowledge domains.

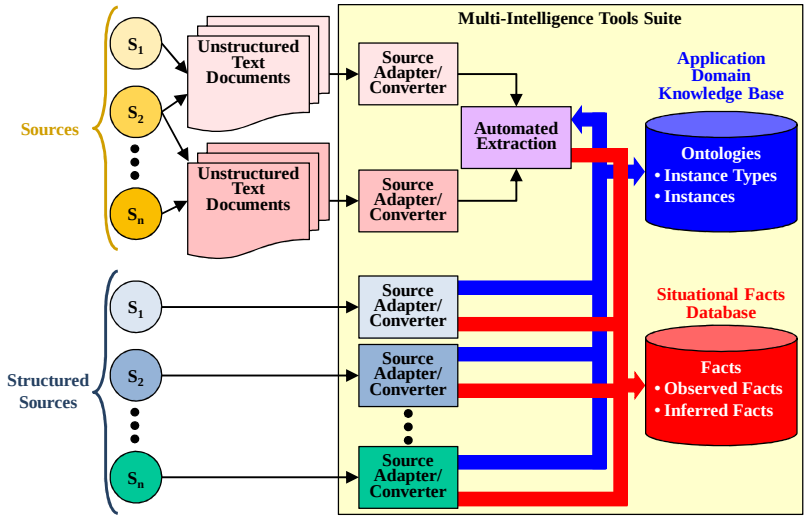

Figure 7. Ingesting data into the MITS

Unstructured documents include all types of documents which contain free text, like Word documents, PDF documents, text documents, emails, etc. To be processed by the MITS system, those documents must first be uploaded one by one, or with one of the massive upload solutions, to the document repository. Text documents can be uploaded from the document manager interface of the MITS. When one has a large amount of documents to upload, it is better to use the FTP solution. Any FTP client can be used to upload documents to the document content repository. Usually, FTP clients will take care of resuming uploads on failure or if the server has to reboot. This is the main reason why it is preferable to use FTP for massive file transfers. Also it is quicker than using the HTTP protocol. The only limitation is that one cannot modify the file metadata from the FTP client. It must be done through the MITS interface after the files have been uploaded.

The MITS document repository (based on Alfresco) exposes its content as a shared folder by creating a virtual computer on the network. This shared folder is presented as if one was browsing its computer folders in a Windows Explorer window. From the shared folder one can open, create, copy, paste, delete, and move documents. All these actions can be done by doing exactly what one would do in a standard Windows Explorer window.

5. Overview of the MITS Main Components

The current version of the MITS provides the intelligence analyst with a variety of innovative features that supports his/her activities and that have already been proven to enhance the resulting intelligence products. In a nutshell, the MITS system provides:

- natural language processing capabilities to support automated semantic analysis of unstructured documents
- automated place name disambiguation and geo-referencing of any piece of information
- semantic and geospatial search for information in sources
- automated entity extraction
- automated collation of all entities of interest pertaining to the knowledge domain(s) of interest
    - person names, date and time elements, locations, organizations, components, effects, triggering mechanisms, types, etc.
- automated/manual fact extraction capabilities from observations contained in sources (published intelligence products in general)
- automated reasoning capabilities over facts observed in sources
- support to trend and pattern analysis
- automated alerting/notification capabilities

Lessons learned from operations in the field show that innovative technologies such as the MITS could support analysts with their threat assessment tasks and could contribute to better defeat hostile activities. Timely targeting of such hostile activities requires the analysis of several, dissimilar sources of intelligence in order to:

- correlate and cross-reference the information available
- “connect the dots” from apparently heterogeneous snippets of information
- seek for trends, patterns, and facts of interest through events reported
- issue timely Indications and Warnings (IW) to the Chain of Command.

Threat assessment requires technology capable to assist analysts in their task by automating, as much as possible, some of the non-analytical and time consuming tasks required (e.g., the manual gathering and collation of large volumes of information from all available sources). Since hostile events are increasing in number, available information also goes increasingly. On one hand, when facing information overload situations, existing systems require more and more analysts, and such human resources might not be available. On the other hand, mature technologies exist that can achieve automated semantic analysis, concept and facts extraction, and trend analysis. Such mature capabilities are available in the MITS system.

The following is a brief, high-level overview of the main components of the current version of the MITS:

- Structured data/information import: Process structured documents. One adapter/converter per type of structured document/source must be developed. The documents are then processed by these adapter/converters such that their content and metadata can be inserted into the MITS knowledge base. Usually, the structured source processing generates facts or modifies the knowledge domains.
    - Track modeling: Can be used to reduce/simplify the track data prior to the conversion of this data into facts.

- Data/information preview: Allows a visual inspection of the data/information prior to their importation and conversion into facts.
- Data/information to fact conversion: Convert the input data/information into facts.

- Unstructured text documents processing: Process new or updated documents received from the document repository service, annotate them, and generate facts according to the text-based templates.
  - Document repository/management: The set of computer programs used to track and store electronic documents and/or images of paper documents. Used to manage documents and notify the text processing module of new documents, document updates and deletion so that they can be processed. The document repository can be accessed by FTP, shared folder or from the MITS user interface. The MITS is typically used to process large amount of documents. All these documents must be stored so that they can be accessed at any time by the user or the system itself for visualization or further manipulations.
  - Automated semantic analysis: This includes text annotation (pieces of text within documents that correspond to a knowledge entity or a pattern of words/tokens defined by an instance type pattern or a text-based template), fact extraction (according to text-based templates), and geo-referencing.
  - Document viewer: Once documents have been processed, they can be viewed in the document viewer. Allow the user to visualize text annotations with their knowledge domain and type, visualize fact annotations, show/hide annotations, include/exclude generic domains in the knowledge domain density calculation, search for text content, show geo-referenced annotations in the GIS (Geographic Information System) window (zoom & center, center, outline), show geo-associations in the GIS, view the document metadata, view the original document, download the original document, open the GIS window manually.
- Statistical analysis of annotations / trend analysis: Analyze occurrences of a certain domain knowledge concept among a set of processed documents that matches multiple user selected criteria. The trend query builder is where one selects the target concept that one wants to exploit and add criteria to refine the analysis. The top trend report contains the query results. One can view a textual description of the query, explaining every selected criterion in details. The query occurrences results are shown with the occurrence count and ratio in percent compared to every occurrence matching the criteria. One can view at a glance the occurrence ratio in a pie chart or a bar chart.
- Automated reasoning / rule-based inference: Deduce fact from other facts by using inference rules (gather facts, apply the inference rules on them and generate new facts whenever a rule prerequisite is fulfilled).
- Subject lists processing: This includes list-based situation monitoring (generate a fact when a subject on a given list is found in the current situation), and situation-based list filling (insert a subject on a list when this subject is found in the current situation and it meets some user-defined criteria).
- Fact management: Allows performing fact search against the MITS fact knowledge base, to visualize facts, as well as to manually insert new facts.
  - Fact search: Execute, save, and load a fact search query. The query can include an atom definition list, a related ontology entities list, geospatial restrictions, validity date range restrictions, a fact type restriction, and a data source restriction.
  - Fact viewer: Allow to visualize facts from the fact knowledge base in tabular format that supports fact sorting/filtering and other features.
  - Fact export: Allow to export some given categories of facts (e.g., facts related to vessel tracks in the maritime domain) in KML format.
- Personalized user notification: Notifications are messages dedicated to a specific user, warning this user of precise events occurring within the MITS system, referencing information or assets he/she is interested in. The module collects the type of information that each analyst is interested in and monitors the system assets. Each time a new asset is added to the MITS knowledge base, the notification module notifies each analyst interested in this asset. Notifications can be displayed as popups, or consulted from the notification manager. If the analyst is connected to the MITS, he/she will receive notifications, independently of which module he/she is currently working with. Actually, one can review its personal notifications at anytime as the MITS keeps every notification, even those generated while the particular analyst is not connected.
  - Notification subscription: Determine who wants to be notified of precise events occurring in the MITS system. Each user has a personalized set of notification subscriptions. A user can be notified of documents containing selected knowledge entities, he/she can also be notified of new facts or workflow states.

- Fact notification: Warn the user that a fact that he/she has subscribed to has been added to the MITS system.
- Document notification: Warn the user that a document containing entities that the user subscribed to has been processed by the MITS system
- Notification management: Allow to manage a personal notification folder, move a notification, view the details of a notification, refresh the personal notification list, mark a notification as read/unread, and remove a notification.

- Knowledge engineering: Provide a user-friendly interface that make the definition, specification and exploitation of atom definitions, built-in functions, inference rules, text-based templates, ontologies, etc. easy for the knowledge engineer and/or the analyst.
- GIS client: Display the geo-referenced annotations, facts, notifications, events and other entities.
  - Area manager: Select and define custom areas, public areas and user defined areas for various uses. It is employed in many modules of the MITS. Each time an area must be selected or defined, the area manager popup is presented to the user.
- Administration: Give access to administrative functions like user management, inference engine execution management, bulk Alexandria data import, and information about the server load.
  - User management: Manage MITS users, as well as their access to Alfresco (the underlying software supporting the document management features of the MITS).
  - Inference engine execution management: Allow to start/suspend/resume/restart the system inference, edit the system inference parameters, and consult the system inference log.
  - Alexandria DB import: The Alexandria database is a geospatial database containing a huge amount of geospatial locations such as countries, capitals, states and regions, etc. One of the steps performed when a new document is sent to the document processing module is to search for any occurrence of the Alexandria database features. For every feature occurrence found, the corresponding feature is automatically imported in the « Geospatial » system ontology as an instance. However, one may want to import Alexandria features in the « Geospatial » ontology to be able to use them elsewhere (in text-based templates, inference rules, etc.) without having to process documents containing these features first.
  - Server load: See how many requests were actually queued in the system and keep an history of the server load such that an administrator can tweak the processing resources configuration.
- Online help.
- User login: username and password

6. MITS Exploitation Example

The best way to appreciate the various features of the MITS is to walk through a practical information and knowledge exploitation example. The one presented in this section involves a totally fictitious drug smuggling scenario in the maritime domain. This scenario has been created for demonstration purpose only, with the intent to showcase a maximum number of components of the current version of the MITS. In this example, the initial trigger for the analysis is a rather vague input from a source:

- There's going to be a drug related event involving a ship
- Expected to happen in some (rather large) area at sea
- In a given time window
- Involving a person listed on a list of suspects

The analysis problem amounts to finding a vessel that:

- Has a historical record of drug smuggling
- Is associated with a suspect person designated by Agency X on a list
- Is on a list of suspect vessels provided by external Agency Y
- Is owned by a person who currently has serious financial problems

- Is within the area where the drug smuggling event is expected to happen in the given time window

Figure 8 illustrates how different components of the MITS are used in this example to process the input data and information made available from the sources (shown on the left hand side of Fig. 8) in order to generate the expected result, i.e., the alert notification to the analyst (shown at the bottom right of Fig. 8).

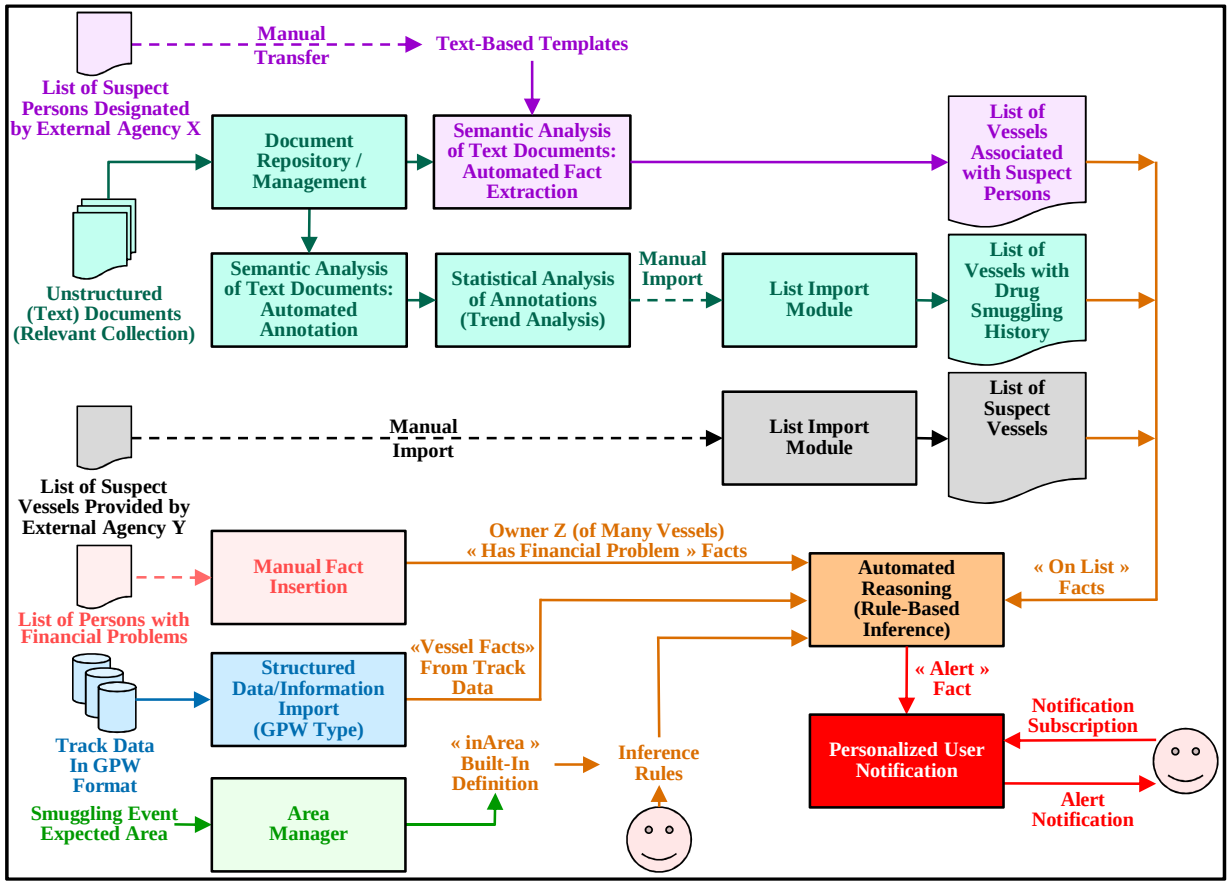

Figure 8. Overview of the MITS exploitation example

Figure 9 illustrates how the analysis ultimately amounts to finding a vessel at the intersection of five lists (the vessel of interest is the *MORDICUS* in this fictitious scenario), which seems to be rather easy to achieve. However, it is very important to note at this point that among the five lists shown on the left portion of Fig 9, only the *List of Suspect Vessels* is actually provided to the analyst as « raw data » from an external agency; the other four lists are created on the fly during the analysis process, by the MITS system itself or the analyst, through the exploitation of all the input data and information provided and using the processing components of the MITS.

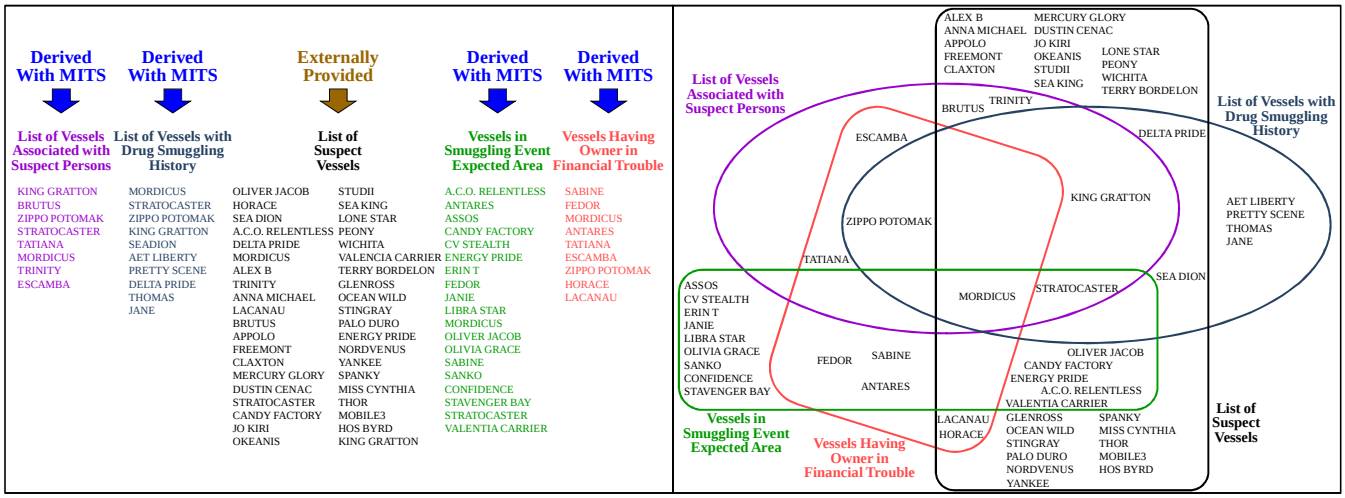

Figure 9. The analysis ultimately amounts to finding the intersection of five lists

Figure 10 shows the three inference rules that are used in this MITS exploitation example to ultimately generate the « *Alert* » conclusion. Rule #1 is used to find vessels at the intersection of three relevant intermediate lists, and to declare these vessels as members of a « *List of Common Vessels* ». Rule #2 is used to generate facts regarding vessels for which the owner is inferred to have financial problems (from the two facts used as the premises of the rule). And finally, Rule #3 will generate an « *Alert* » fact whenever the position of a vessel is in the area of interest, the owner of this vessel has financial problems, and this vessel has been inferred to be a member of the « *List of Common Vessels* ». The MITS exploitation example provided in this paper illustrates how these three inference rules are actually created on the fly by the analyst, as required by the analysis problem at hand, using the knowledge engineering module of the MITS.

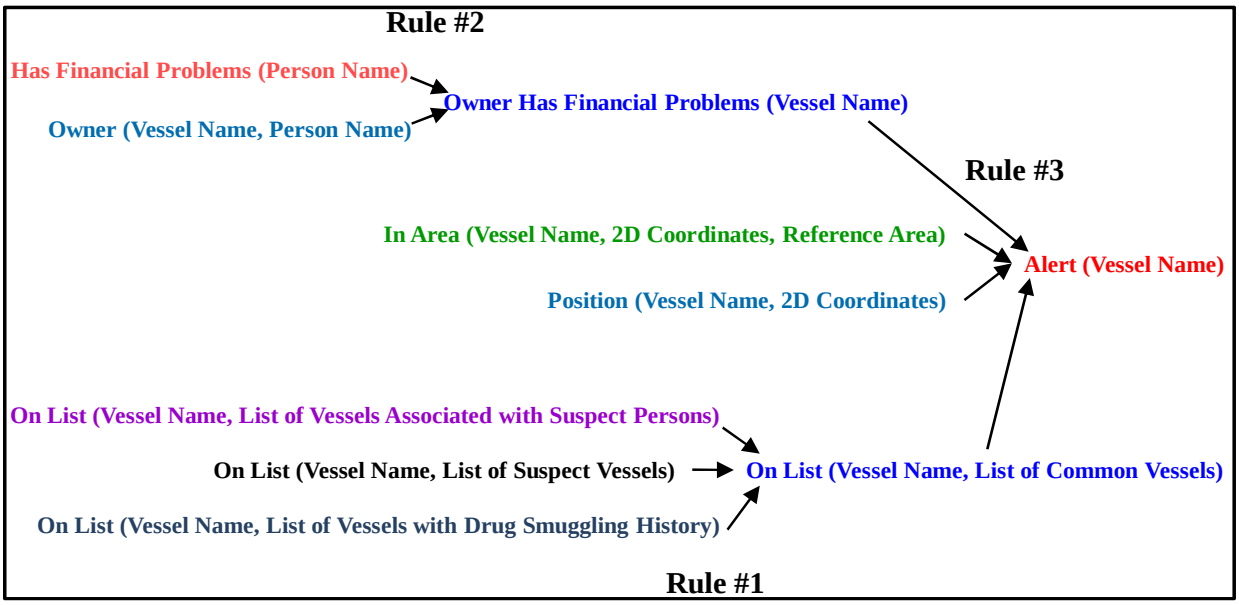

Figure 10. Forward chaining of the inference rules for the MITS exploitation example

Table 1 provides a detailed walk through of the MITS exploitation example.

Table 1 – Walking Through the MITS Exploitation Example

To Exploit	Analyst Activities/Steps with the MITS
• List of suspect vessels designated by external Agency Y	• Create a new « List of Suspect Vessels » • Select the subject type as « Vessel Name » • Manually import the elements of the list provided by Agency Y into the list created above to generate « On List » facts and potentially augment the domain knowledge ontology with new vessel names
• List of suspect persons designated by external Agency X • Relevant collection of unstructured text documents	• Create an empty « List of Vessels Associated with Suspect Persons » • Create the Text-Based Templates (one per suspect person) with an « On List » atom (fact) in the conclusion • Upload the text documents from the collection to potentially generate « On List » facts • Verify the resulting list using the List Export module
• Relevant collection of unstructured text documents	• Uploading the text documents in the previous step automatically generated domain-related annotations • Visualize the annotations using the Document Viewer module • Using the Trend Analysis module to statistically analyze the resulting annotations, create a relevant query and generate an analysis report • Create a new « List of Vessels with Drug Smuggling History » • Manually import the query results into the list created above to generate « On List » facts
• « On List » facts (generated by the creation of the different lists in the previous steps)	• Create an empty « List of Common Vessels » • Create the inference rule #1 with three « On list » atoms (facts) as the premises, and one « On List » atom (fact) in the conclusion
• The information on (relevant) persons having financial problems	• Create an « Has Financial Problems » atom having the « Person Name » as an argument, with potentially other arguments that could also be useful to describe the facts

To Exploit	Analyst Activities/Steps with the MITS
	• Manually insert « Has Financial Problems » facts using the Manual Fact Insertion module
• « Has Financial Problems » facts (manually inserted) • « Owner » facts (from GPW track data)	• Create an « Owner Has Financial Problems » atom having the « Vessel Name » as an argument, with potentially other arguments that could also be useful to describe the facts • Create the inference rule #2 with the « Has Financial Problems » and « Owner » atoms (facts) as the premises, and an « Owner Has Financial Problems » atom (fact) in the conclusion
• The information on the expected area where the drug smuggling event is expected to happen • « Owner Has Financial Problem » facts • « Position » facts (from GPW track data) • « On List » facts (i.e., the List of Common Vessels)	• Create an « Alert » atom having a « Vessel Name » as an argument, with potentially other arguments that could also be useful to describe the alert • Create the inference rule #3 with the « inArea » built-in definition (configured with the appropriate relevant area by using the Area Manager of the GIS module to create the appropriate polygon), the « Position », « On List » and « Owner Has Financial Problem » atoms (facts) as the premises, and the « Alert » atom (fact) in the conclusion
• The « Alert » fact	• Create a notification subscription on the « Alert » atom (fact)
• Maritime track data set (in GPW format)	• Use the « GPW Data Import » module to: − Select the track modeling algorithm − Select the original source − Select the date range (time window) − Select the region (spatial window) − Select which facts shall be generated during the import − Inspect vessel tracks in GoogleEarth prior to the import − Launch the data import
• All of the above	• Configure the inference system of the MITS (in the Administration module) • Launch the inference process
• All of the above	• Monitor the « Alert » notification(s) • Acknowledge the notification(s) • Further investigate the notification to increase the understanding of the situation − View the justification of the Alert
• All of the above	• Use the Fact Viewer of the Fact Management module to explore the different facts • Export the facts in KML format • Use GoogleEarth to visualize the track data and the corresponding facts

7. MITS Implementation and Evolution

The MITS has been created in 2008 and it has since evolved in different directions, following research opportunities sponsored by different customers/partners. One aspect that has been studied in depth and implemented in the MITS is automated reasoning. The rule-based inference capability described above in this paper has been evolved into a broader framework enabling the inference of situational facts through the synergistic exploitation of the complementary strengths and expressiveness characteristics of different knowledge representation approaches, and corresponding reasoning paradigms [Roy, 2009]. In this framework, automated reasoning has been implemented as a set of independent modules performing rule-based [Roy, 2010], description logic [Roy, Davenport, 2010] and case-based [Bergeron Guyard, Roy, 2009] reasoning. A module for kinematics and geospatial reasoning has also been implemented as specific, dedicated inference code [Roy, 2009].

Clearly the MITS will continue to evolve as new desirable functionalities are identified by the operational intelligence community, and also in order to integrate legacy intelligence analysis support tools that have not yet been included in any prior version of the MITS. As a result of the experience gained through the incremental development of the MITS so far, a number of lessons learned have been formulated regarding design and implementation approaches, enabling technologies, project and system development governance, etc. Among these, the need for a more suitable integration platform and the evolution towards a different human-computer interaction layer are briefly discussed next.

7.1 Intelligence S&T Integration Platform (ISTIP)

The efficient development of a suite of integrated tools like the MITS also brings forward some integration requirements and issues at the software system level. On one hand, from an “intelligence support tool” perspective, each KIME tool in the MITS federation must be autonomous and self-protected, it must own and control information in accordance with some local

policies, and it must have boundary protection devices to enforce such local policies. Different federated mechanisms are required to facilitate knowledge/information/data (KID) exchange and machine-to-machine collaboration in this federation of KIME tools. On the other hand, from a research and development (R&D) perspective, the I&I Section has a need for a software system development platform that meets a number of requirements. First, the platform has to be suitable for research activities. In a research organization, the system developers must have the flexibility to use any technology (especially the emerging ones) to implement the KIME tool prototypes, without being always constrained by the technological choices previously made for the baseline systems of the customers/partners from the operational community. Second, the platform must support the incremental development of components and capabilities when going from one project to the next, such that one doesn't have to re-invent the wheel each time a new research activity is started. With an incremental development approach, new projects can be proposed that already have access to a core of capabilities previously developed in prior projects. Third, as much as possible, the platform shall not require an overarching authority among the tool developers. The introduction of any new KIME tool shall be negotiable among the tool developers, and these developers shall be free to select appropriate mechanisms to meet their ambitions and to enforce local policies. Finally, the platform shall be scalable as new KIME tools are progressively added to the pool of tools.

In response to the requirements briefly outlined above, the I&I Section at DRDC Valcartier has proposed and developed the Intelligence S&T Integration Platform (ISTIP). Using open source Web services technologies and service-oriented architecture (SOA) design principles, the ISTIP provides a backbone, integration reference platform for the iterative and incremental development and integration of the innovative, loosely coupled, reusable, composable and interoperable services required to perform tasks in computer-based intelligence support systems.

From now on, the ISTIP will be the backbone of the MITS. As much as possible, each KIME tool of the MITS will be developed as a set of services that meet a number of SOA design principles:

- Loose Coupling – Services maintain a relationship that minimizes dependencies and only requires that they retain an awareness of each other.
- Service contract – Services adhere to a communications agreement, as defined collectively by one or more service descriptions and related documents.
- Autonomy – Services have control over the logic they encapsulate.
- Abstraction – Beyond what is described in the service contract, services hide logic from the outside world.
- Reusability – Logic is divided into services with the intention of promoting reuse.
- Composability – Collections of services can be coordinated and assembled to form composite services.
- Statelessness – Services minimize retaining information specific to an activity.
- Discoverability – Services are designed to be outwardly descriptive so that they can be found and assessed via available discovery mechanisms.

At this moment, a number of services aligned with the principles described above have already been implemented and are available on the ISTIP. Examples are the Inference of Situational Facts through Automated Reasoning (ISFAR), the Semantic Annotation of Text Documents (SATD), Spatial Features Management (SPFTM), Track Modelling (TM), GPW Tracks to Situational Fact Knowledge Base (GPWT2SFKB), Ontology Repository (OR), and Document Repository (DR). Other services already exist, and many more will continuously be created as legacy tools and systems are converted, and as the research activities of the I&I Section at Valcartier further progress.

7.2 Visionary Overarching Interaction Interface Layer for the Analyst (VOiiLA)

Clearly, providing all of the services of the ISTIP is not sufficient to achieve the concept of the MITS. One also needs some human-computer interaction front-end for the exploitation of these services. This front-end is called *VOiiLA* (Visionary Overarching Interaction Interface Layer for the Analyst). All services of the ISTIP combined with VOiiLA constitute the latest version of the MITS.

8. Conclusion

This paper discussed the MITS from different perspectives, presenting it as a key component supporting research and development in information and knowledge exploitation in the intelligence domain. The main characteristics of the MITS were reviewed. The central notions of domain knowledge and situational facts were discussed, along with the ingestion in the MITS of structured and unstructured data and information. The main modules of the MITS were briefly described, and a detailed exploitation example highlighting some of its powerful and innovative capabilities was provided. The SOA platform and human-computer interaction components that together constitutes the MITS were introduced.

The MITS has been created in 2008 and it has since evolved in different directions, following research opportunities sponsored by different customers/partners. Clearly the MITS will continue to evolve as new desirable functionalities are identified by the operational intelligence community, and also in order to integrate legacy intelligence analysis support tools that have not yet been included in any prior version of the MITS. From the experience gained through the incremental development of the MITS so far, a number of lessons learned have been formulated regarding design and implementation approaches, enabling technologies, project and system development governance, etc. From now on, as a result of some of these lessons, the development of the MITS will build on the ISTIP (a more suitable integration platform) and VoiiLA for the human-computer interaction layer.

9. References

[Auger, 2009], Auger, A., *Acquisition and Exploitation of Knowledge for Defence and Security*, NATO IST-087 Symposium on Information Management / Exploitation, Stockhlom, Sweden, 19-20 October 2009.

[Bergeron Guyard, Roy, 2009], Bergeron Guyard, A. and Roy, J., *Towards Case-Based Reasoning for Maritime Anomaly Detection: A Positioning Paper*, Proceeding of The IASTED International Conference on Intelligent Systems and Control, Cambridge, Massachusetts, USA, 4-6 November, 2009.

[Boury-Brisset, 2001], Boury-Brisset, A.-C., *Towards a Knowledge Server to Support the Situation Analysis Process*, Proceedings of the Fourth International Conference on Information Fusion (FUSION 2001), Montreal, Canada, August 7-10, 2001.

[Brachman, Levesque, 2004], Brachman, R. J. and Levesque, H. J., *Knowledge Representation and Reasoning*, Elsevier, Morgan Kaufmann, San Francisco, CA, 2004.

[DMR, 2010], DMR, *Customization of the MITS Technology for the Counter-IED Task Force – MITS 2.2 User Guide*, Version 1.0, Fujitsu Consulting (Canada) Inc., DRDC Valcartier Contractor Report, Scientific Authority: Dr Alain Auger, 24 November 2010.

[Giarratano, Riley, 1998], Giarratano, J. and Riley, G., *Expert Systems - Principles and Programming*, PWS Publishing Company, Boston, 1998.

[Ginsberg, 1993], Ginsberg, M., *Essentials of Artificial Intelligence*, Morgan Kaufmann, San Francisco, California, USA, 1993.

[Gómez-Pérez et al, 2004], Gómez-Pérez, A., Fernández-López, M. and Corcho, O., *Ontological Engineering*, Springer, London, 2004.

[Gruber, 1993], Gruber, T. R., *A Translation Approach to Portable Ontology Specification*, Knowledge Acquisition 5(2): 199-220, 1993.

[McIntyre et al, 2003], McIntyre, S.G., Gauvin, M. and Waruszynski, B., Knowledge Management in the Military Context, Canadian Military Journal, Spring 2003.

[Merriam-Webster, 2003], *Merriam-Webster's 11th Collegiate Dictionary*, Software, Version 3.0, 2003.

[Roy, 2001], Roy, J., *From Data Fusion to Situation Analysis*, Proceedings of the Fourth International Conference on Information Fusion (FUSION 2001), Montreal, Canada, August 7-10, 2001.

[Roy, 2006], Roy, J., *Combining Elements of Information Fusion and Knowledge-Based Systems to Support Situation Analysis*, Proceedings of Multisensor, Multisource Information Fusion: Architectures, Algorithms, and Applications 2006, SPIE Symposium on Defense and Security, Orlando, Florida, April 17-21, 2006.

[Roy, 2007-A], Roy, J., *A Knowledge-Centric View of Situation Analysis Support Systems*, DRDC Valcartier TR2005-419, January 2007.

[Roy, 2007-B], Roy, J., *Holistic Approach and Framework for the Building of Knowledge-Based Situation Analysis Support Systems*, DRDC Valcartier TR2005-420, January 2007.

[Roy, 2009], Roy, J., *Automated Reasoning for Maritime Anomaly Detection*, Proceedings of the NATO Workshop on Data Fusion and Anomaly Detection for Maritime Situational Awareness (MSA 2009), NATO Undersea Research Centre (NURC), La Spezia, Italy, 15-17 September 2009.

[Roy, 2010], Roy, J., *Rule-Based Expert System for Maritime Anomaly Detection*, Proceedings of Sensors, and Command, Control, Communications, and Intelligence (C3I) Technologies for Homeland Security and Homeland Defense VIX, SPIE Defense, Security, and Sensing 2010, Paper 7666-97, Orlando, FL, USA, 5 - 9 April 2010.

[Roy, Auger, 2008-A], Roy, J. and Auger, A., *Knowledge Representation Concepts, Paradigms and Techniques for Use in Knowledge-Based Situation Analysis Support Systems*, DRDC Valcartier TM2006-755, octobre 2008.

[Roy, Auger, 2008-B], Roy, J. and Auger, A., *Reasoning Processes, Methods and Systems for Use in Knowledge-Based Situation Analysis Support Systems*, DRDC Valcartier TM2006-756, octobre 2008.

[Roy, Auger, 2008-C], Roy, J. and Auger, A., *Knowledge and Ontological Engineering Techniques for Use in Developing Knowledge-Based Situation Analysis Support Systems*, DRDC Valcartier TM2006-757, octobre 2008.

[Roy, Davenport, 2010], Roy, J. and Davenport, M., *Exploitation of Maritime Domain Ontologies for Anomaly Detection and Threat Analysis*, 2nd International Conference on Waterside Security (WSS 2010), Marina di Carrara, Italy, 3-5 November 2010.

[Russell, Norvig, 1995], Russell, S. and Norvig, P., *Artificial Intelligence - A Modern Approach*, Prentice Hall, New Jersey, 1995. [Stefik, 1995], Stefik, M., *Introduction to Knowledge Systems*, Morgan Kaufmann Publishers, San Francisco, California, 1995.

[Stefik, 1995], Stefik, M., *Introduction to Knowledge Systems*, Morgan Kaufmann Publishers, San Francisco, California, 1995.

[Struder et al, 1998], Struder R., Benjamins, V. R. and Fensel, D., *Knowledge Engineering: Principles and Methods*, IEEE Transactions on Data and Knowledge Engineering 25(1-2): 161-197, 1998.

[Turban, Aronson, 1998], Turban, E. and Aronson, J. E., *Decision Support Systems and Intelligent Systems*, Fifth Edition, Prentice Hall, New Jersey, 1998.

[Waltz, 2003], Waltz, E., *Knowledge Management in the Intelligence Enterprise*, Artech House, Norwood, MA, 2003.