15th ICCRTS
"The Evolution of C2"

Title: Human and Machine interaction with Knowledge Bases

Topic:
Topic 3: Information Sharing and Collaboration Processes and Behaviours
Topic 2: Networks and Networking

Author:
Simon Bray, Dstl

Point of Contact:
Simon Bray
Defence Science and Technology Laboratory (Dstl)

Information Management Department
Rm C003, Grenville Building
Dstl Portsdown West
Fareham
Hants
PO17 6AD
United Kingdom

Tel:
+44 (0) 2392 537851

E-mail:
sebray@dstl.gov.uk

Dstl reference number: Dstl/CP40411

Title: Human and Machine interaction with Knowledge-Bases.

Author: S E Bray, Dstl. sebray@dstl.gov.uk

Abstract: A philosophical basis for constructing knowledge-bases is established and the terms "knowledge", "information" and "facts" defined in relation to it. It is then proposed how knowledge should be represented in knowledge-bases, and the need explained for a common language for machine to knowledge-base interaction, and machine-to- machine interaction about knowledge. An approach to the development of this common-language is provided. Human interactions are not intended to use this common-language directly, and it is explained how different languages used by humans and external systems can be integrated into a knowledge-based system. Language is discussed in terms of its semantic and pragmatic components, showing how different categories of information exchange session can be used to standardise the pragmatic components such that the intended meaning is clear beyond the literal meaning; including, critically, how the knowledge-base contents should change after each interaction, and how to respond (where appropriate).

1. Introduction

The context of this work is a research project being undertaken by the author at the UK Defence Science and Technology Laboratory, which is developing the concept of a Virtual Knowledge Base (VKB) to support the Warfighter. The scope of the VKB concept is illustrated below. A VKB will consist of Information, "Known-facts" (defined below), and Warfighter Information Services – all distributed across a network. Several categories of Warfighter Information Service have been identified including services to support Information Extraction, Information Collation and Validation, Situation/Intent Assessment, and services that use information and known-facts generated from the above services to directly support the Warfighter – enabling better Situational Awareness and Decision Support.

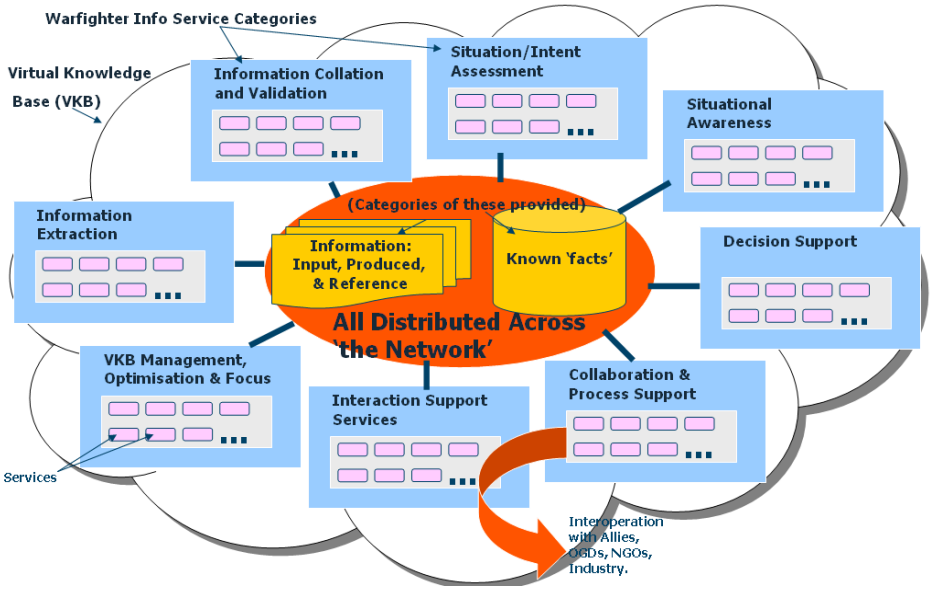

Figure 1 Virtual Knowledge Base concept and Warfighter Information Service Categories

This paper focuses on one aspect of the design of such a Virtual Knowledge Base - language: considering both the internal formal-logic language used to represent "known-facts" in a knowledge-base and languages used for interactions with a knowledge-base through services. The central thrust of this work is to enable machines to reason over all that is known in computers connected to a

network, from any Information presented in any format or language, to support Situational-Awareness and Decision-making.

2. Facts, Information and Knowledge

The terms "fact", "information" and "knowledge" do not have widely accepted meanings – and have been the subject of disagreement among philosophers for millennia. To make progress in the construction of knowledge-based systems it is necessary to select a particular philosophical basis for discussion, and then to define these terms with respect to that philosophical outlook. The philosophical basis chosen here draws on ideas first proposed by Wittgenstein (facts)[1], Locke (ideas vs. words)[2], Kant (categories of judgement)[3] and Searle (meaning of expressions)[4]. The ideas expressed here underpin many computer-based approaches in use today including the Universal Modelling Language (UML)[1], data-models such as the NATO JC3IEDM and the W3C's Semantic-Web initiative. Having a symbolic representation of facts, following Wittgenstein's approach, is recognised as being important for Information Fusion [6]. Figure 2 shows the distinction between the basic ideas.

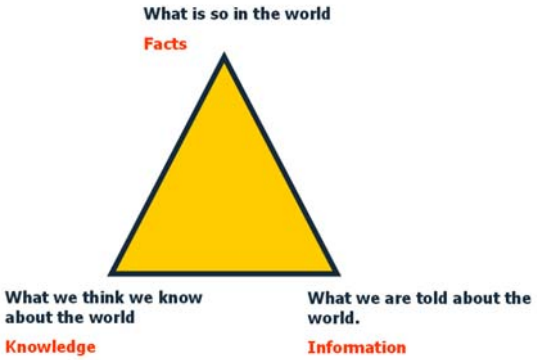

Figure 2 – Facts, Information and Knowledge

Facts are what is so in the world [1]. The short phrase "is so" encompasses a vast amount: including what physically exists in the world, and what "is so" among what exists in the world, and what notions 'exist' in the minds of people in the world, including their beliefs and intentions. In this paper the distinction between the physical-world and what exists in the minds of people in the world is made by calling the former the "Objective-World" and the latter the "Subjective-World". Worlds are discussed again later, and two other kinds of world introduced.

Information is used for different purposes (described more fully below), but the most common one - unsurprisingly, as its name implies - is to *inform*, i.e. to tell us something about the world. This may be done in many different languages (Natural Languages (English, French ...), Computer-Orientated Languages, and non-textual, symbolic languages). A distinction is made between physical objects that contain information (e.g. a document or message), which are here called "information-objects", and the information itself which is logical, i.e. the information contained in an information-object is the meaning of its contents.

Knowledge: On the basis of information received, prior-knowledge, and the capacity to reason about what we are being told, to judge it, and to figure out how new information relates to what was previously known, we come to a new understanding of the world – expressed in the above diagram as "what we think we know about the world". In short-hand, this amounts to *known-'facts'*, where the

[1] The link between Wittgenstein and UML is made explicitly in ref [5].

term ‘facts’ is in quotation marks because is not certain that these *known-‘facts’* correspond with the facts as they are in the real-world. (It will also be explained elsewhere that such *known-‘facts’* can be known to be uncertain, inaccurate or conflicting and be recorded as alternate hypotheses, or on a probabilistic basis, or with an assessed error-margin.)

But knowledge does not only consist in knowing ‘facts’: the other aspect is “know-how”, i.e. knowing both how to discern what the facts are, but also how to use what is known to some purpose. *The acid test of having knowledge is the ability to make appropriate decisions.* The degree of understanding of the world necessary to make decisions depends on the nature of the decision, and knowing-beings use internal “conceptual-models” as the basis for recording and reasoning about their understanding of the world. (Humans also use variations of these conceptual-models for perception.) It is considered that both people and software-agents can be counted as “knowing beings”.

A particular kind of conceptual-model is here proposed as the basis for VKB development: this is the Entity-Attribute-Relation model. This model consists of Entity-classes, each of which may have certain Attributes (and values of the Attributes), and each Entity-Class may have certain Relations with other Entity-Classes. The *known-‘facts’* of what is believed to exist and be so among what exists are then recorded about and between Entities that are *instances* of the Entity-Classes in the conceptual-model. Known-facts are represented as logical “propositions” within a conceptual-model of the world, whereas the Information contained in Information-Objects that are input-output may be expressed in many languages, as illustrated below.

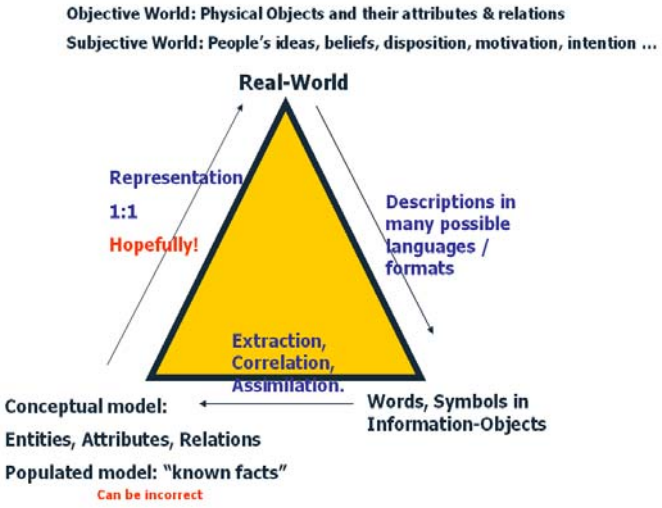

Figure 3 – Detailed view of Facts, Information and Knowledge

The key point is that *known-‘facts’* are an abstraction within a conceptual-model, and that these known-facts are independent of any language that was used to provide the Information about the world from which the known-facts were derived (by a complex process of extraction, correlation, assessment and assimilation/inference). Computer programmers may well choose to give these conceptual entities labels which are human understandable in a natural language, but what a programmer understands by a label is not relevant to their use by machines for reasoning purposes. Machines do not “understand” the labels, they are just identifiers used to distinguish one concept from another. They could just as well be numbers. In fact a very good precedent for using numbers for concepts is provided by Roget’s Thesaurus [7] which presents a schema of 999 “general ideas” (i.e. concepts), each idea being assigned a number 1-999. His thesaurus then provided a mapping of words and phrases (in a particular natural language, originally English) to each concept-number. Having English and French versions of Roget’s Thesaurus would thus enable Information presented in different languages to be abstracted (fused) into

a common conceptual model, each concept being assigned a unique number which is independent of any language.

This abstraction to a conceptual-level is what enables machines to reason over all that is known in a network (subject to information-access controls), from any Information presented in any format or language, to support Situational-Awareness and Decision-making. It enables Querying, Question-Answering, and automated reporting of what is known (both on-demand, and if required, when certain criteria are met that trigger an output). This offers a revolutionary change in capability to exploit information to military effect.

In a VKB this new capability of machine reasoning about known facts is as well as not instead of, traditional methods of information storage, search and retrieval through word-searches and meta-data. Both capabilities stand side-by-side, and can be accessed by users as required. This is illustrated in the figure below.

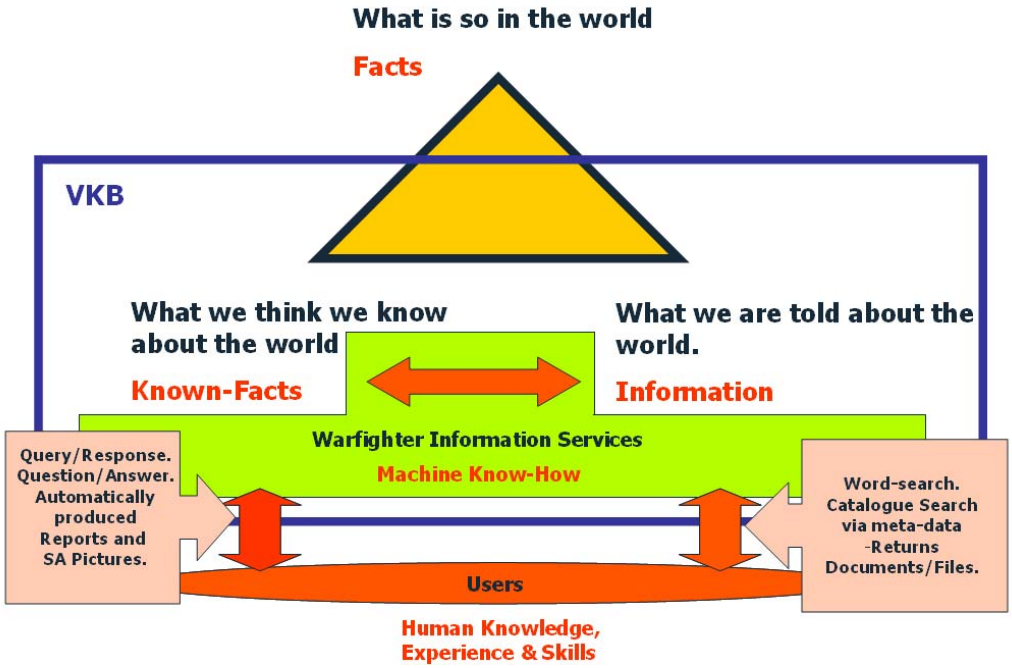

Figure 4 – Scope of VKB and its Interactions with Humans

3. Query/Response vs. Question/Answer

Queries are used to interrogate data-bases and the response contains sets of records extracted from the data-base that match the specified selection criteria. The queries contain field-names as part of the selection criteria, and therefore the person or machine creating the query has to know the structure of the data-base. What is returned is limited to what is contained in the data-base, or some very simple numeric functions over these contents (e.g. total, count).

Questions are used to interrogate knowledge-bases and return answers to the questions. The person or machine forming the question need have no knowledge of how information is stored in the knowledge-base and he/he/it may phrase the question in a Natural Language or a Controlled Natural Language, and get responses in the same form. Answering a question is much more complex than responding to a query, though part of the process of question answering may be the automated formation of data-base queries and the synthesis of their responses into an answer to the question.

For this to be possible at all there needs to be in place an appropriate schema for the representation of knowledge relevant to the subject of the question, and this schema must to some extent be shared by the question-poser and question-responder. (This is not the same as knowing what the "field names" are in any supporting data-bases.) Furthermore the response mechanism must contain reasoning capability that range over the schema, enabling it to interpret the question in the context of the schema, to map the information contained in supporting data-bases to the schema, and hence to determine which data needs to be retrieved and how it should then be processed to produce the answer.

4. Language

Language is the means by which we encode information to communicate meaning. Meaning is derived both from semantics and pragmatics. Semantics is how the literal meaning of an expression is derived from the vocabulary, syntax and morphology of the language (grammar = syntax + morphology). Pragmatics is how additional meaning (over and above the literal meaning of an expression) is derived from context. Specifically, pragmatics enables the significance an expression to be understood in the context a set of information exchanges and background knowledge. Background knowledge includes training and experience, which shape expectations of what will happen as a consequence of the set of information exchanges (prior, current, and potential future ones).

The difference between semantics and pragmatics is easy to appreciate from the following simple example:
Person A says: XXXX (something)
Person B says: Yes

What does "Yes" mean? The semantics (in this case it is just the vocabulary as there is only one word in the sentence) tells you the literal meaning. A dictionary provides two possible meanings (senses) of "yes": "affirmation" and "consent". It is usual for English words to have more than one sense. How these meanings are disambiguated will be discussed in a moment. But whichever of these meanings was intended there is more entailed in saying "yes" than knowing whether it means affirmation or consent. What is entailed by saying "yes" in this context clearly depends on what A said. This could be trivial or of earth shattering importance, and may result in some action by A or B (or both), affecting A, B and/or others. This entailment is the pragmatic part of the meaning of the utterance.

"Yes" could therefore mean a commitment by B to do something, approval of a proposal by A for A to do something, or it might be the answer to question from A (in which case the meaning of "yes" entails that a proposition made by A is true), or might simply be an acknowledgement of hearing what A said.

In this paper we are concerned both with languages used by humans and languages used by machines (and those that both can understand). It is important to understand how machines work-out the meaning in information-exchanges, both the literal meaning and the meaning in context.
In this example (which is also usually the case) there was ambiguity in the literal meaning of what B said, as well as a range of possible entailments. This is a hard task with Natural Languages, but is much easier with Controlled Natural Languages and "computer-orientated" languages (such as those based on XML or ADatP-3).

The above discussion considered how the meaning of expressions used in information exchanges is to be determined through the semantics of the language used to create the expressions and the pragmatics

of the type of information exchange that is in progress. In this case the aim is for the parties to the information exchange to each understand what the other(s) "meant" by their utterances. This is necessary but is not the end-result that we are aiming at: we also wish to extract new knowledge (known-facts) from information received and store this in a knowledge-base. To do this we need a formal language for knowledge-representation (defined in section 6), and a way of extracting what we want to know from the information that is received (see section 12). Extracting "what we want to know" is a purposeful activity. It is important to recognise that in this case. What the recipient wants to know may not be literally contained in the information received - it may need to be inferred from it. This distinction drawn here is between the "meaning" of information (which is known to the sender) and the "significance" of information which depends on the recipient's interest/purpose (which may or may not be known to the sender). In the context of fact-extraction into a knowledge base it is not an individual recipient who determines the significance or purpose of extraction, it is a community of users who collectively define a knowledge-representation schema (of entities attributes/values, and relations) to record what they want to know (for their joint purposes), and who also define a set of "fact-extraction rules" which define a mapping from information received to recorded known-facts, taking into account the significance of the information for their purposes. Sections 8-12 expand on these points.

5. The development of a common language for Warfighter Information Services

When Warfighter Information Services express information in their information exchanges with other Warfighter Information Services they need a language that is easier for machines to understand than Natural Languages. There have been several initiatives to design such languages, notably:

- Knowledge Interchange Format [8] (DARPA project)
- Common Logic (ISO/IEC IS 24707:2007) [9] (partly derived from KIF)
- The semantic-web (a suite of standards including XML, RDF, RDFS, OWL).

In all these cases it is possible to abstract what is to be expressed as a directed graph (of subjects, predicates and objects; where subjects and objects are graph vertices and predicates are edges), and then to define a way of "serialising" the content of the graph for transmission. This distinction is explicit in the case of the semantic web in that different standards are defined for graph-definition (RDF [10]) and serialisation (usually XML). The next problem to be addressed is how the semantics of the graph are to be defined. In the case of the semantic-web this has been explicitly defined both through "RDF Semantics" [11] and RDF-Schema [12].

KIF, Common-Logic and the semantic-web are all flexible and extensible approaches with broad applicability. When designing a Virtual Knowledge Base with associated Warfighter Information Services we need to choose a set of standards to be used for abstract representation of knowledge and its serialisation for transmission - but we also need to agree among the design community what it is that must be represented about each "atom" of knowledge, and how we express more complex "molecules" of knowledge? The above standards provide the tools for forming these representations and expressions but do not themselves provide the detail that is needed. This lack of necessary detail is a consequence of their general-purpose extensible approach. We now need to pin-down these details in order to build Knowledge-Bases whose contents can be unambiguously understood by humans and machines, yet have a an expressive power that is rich enough to deal with real-world (military) situations.

Regarding the richness of expressive power, in military Virtual Knowledge Bases we want to be able to:

- represent assertions of affirmation and denial of reported facts (that something is, or is not, so)

- state assumptions, hypotheses and plans – and distinguish these from what is actually so;
- distinguish among a set of facts asserted at different times which is the latest;
- distinguish between asserted and authoritatively declared facts (explained in section 6).
- handle ambiguity, uncertainty, probability, and state facts in logical combinations (e.g. that A or B is so, or that A and B are mutually exclusive).
- record the provenance of reported facts, and different views about them (not assuming a "single view of the truth").

There are different ways in which these issues could be handled in RDF, KIF, or Common Logic – what is needed to design a large scale VKB is agreement among the design community on an abstract representation (in this case a directed graph) which can then be serialised by any language of choice - safe in the knowledge that the serialisations are isomorphic (i.e. they can be translated from one serialisation standard to another without loss of meaning). The main purpose of this paper is to propose this abstract representation in order to move towards consensus in the research community, and eventually in the acquisition community.

6. Propositions

The basic "atom" of knowledge proposed here is a "proposition", which is used to represent a simple "known fact". Ways of combining propositions to represent more complex "known facts" are also defined. A proposition is here defined as:

"An *asserted* or *declared* statement whose truth can be affirmed or denied"

The difference between an *assertion* and a *declaration* is that anyone may *assert* something to be so as a result of having observed it to be so, or it having been reported to be so, or formed an opinion that it is so. Whereas, a *declaration* changes the world: it is so because the declarer says it is, and this entails that the declarer has some authority to make this declaration, and so not everyone is permitted to make declarations on a given subject [4].

In the simplest case, a proposition consists of a statement and a set of "qualifier flags" (explained below). A statement here refers to a subject, predicate, and object (as in RDF).

The need to "qualify" statements stored in a knowledge-base to form "fully qualified propositions" arises because there are a number of different ways in which a statement can be made, depending on the context of a conversation (information-exchange session). This was first recognised by Kant [3] who distinguished several different "categories" of judgement for example that a statement may be made assertorically, hypothetically or apodictically. More recently the theories of language-games [13] and "speech acts" [4] have been developed, in which the meaning of statements is affected by the context of their being stated.

In a Virtual Knowledge Base we wish to record "known facts" (propositions) in a manner that explicitly captures the context of statements, in so far as the context affects the meaning. This is because we wish to be able to reason over the known facts without having to keep re-processing the original text to obtain the relevant context.

The following set of six "qualifier-flags" is proposed to record critical distinctions between statements:

- True / False / Possible / Impossible (handles conflicting views of truth)
- Asserted / Declared /Hypothetical (explained above)

- Categorical/ Probabilistic/Inferred (handles uncertainty*)
- Simple / Alternate / Combined (handles ambiguity & logic*)
- Historical/ Latest/ Future/ Defined (handles time)
- Objective / Subjective / Alternative / Universal Worlds (handles frames of reference, including hypotheses/ plans /options)

The idea is that each statement would be qualified with one of the above options for each of the six qualifier-flags. Where a flag is marked with a "*" additional statements would be provided to provide additional information (e.g. the probability value for a probabilistic statement). These statements are predicated on the original statement, and so are here termed "2nd-order propositions". Many other uses of 2nd-order propositions to capture the context of the original statement are defined at Appendix A. The most familiar of these is to record the provenance of a statement.

It is possible to represent these "qualifying flags" and 2nd-order propositions within the RDF language by reifying the original statement, and then predicating the set of flags or the 2nd-order statement on the reified statement. (This detail is mentioned to clarify that the RDF language can provide this functionality; not to suggest that it is essential that RDF be used.)

It is recognised that not all existing knowledge-bases will contain the detail proposed in the set of qualifying-flags and so their "Level of conformance[2]" can be defined by the flags that they support. When reading from such knowledge-bases it is possible to add default values to each missing flag in order to make it conform to the standard (for example: True/False default is True[3]). However interoperability is limited when writing to such knowledge-bases, i.e. if a certain flag is not supported in a knowledge-base then propositions with other than the default value cannot be recorded (and must not be recorded as this would alter their interpretation when read-out).

If we regard an "atom" of knowledge as a "proposition" (at defined above) then we can define "molecules" of knowledge as different kinds of "set". For example we may wish to assert that the logical conjunction of two or more propositions is true (a logical set whose members are combined with the AND operator); or that a set of propositions are mutually exclusive (only one of them is true) (a mutually exclusive set). The capability to do this is more explicitly specified in KIF and Common-Logic than RDF, but again RDF can still be used if we create a suitable schema (in RDF-S) to define the meaning of the "molecular bonds" between (reified) propositions. Details of how this may be done, and a full explanation of the proposed different kinds of "set", are provided at Appendix A.

The above discussion of "atoms" and "molecules" of knowledge (with Appendix A) has proposed what needs to be captured when "knowledge" is stored in a VKB such that it can be interpreted correctly by machines for the purposes of machine reasoning without recourse to re-reading the discourse from which the "known-facts" were extracted to gain the correct context.

But we do not just need a machine-understandable method of storing knowledge; we also need a machine understandable language for interaction between machines and Knowledge-Bases, and between machines and/or humans about knowledge. It is not simply that a method of "serialisation" of sets of propositions is needed (which can be readily achieved by various means including XML). It is also necessary to capture and serialise the "intent" of the information exchange, which will determine what should be done with the propositions provided - including what response is expected and what change should be made to the knowledge-base – which is often much more complex than simply

[2] The idea of "Levels of Conformance" is adopted in KIF.
[3] The full set of default values are True, Asserted, Categorical, Simple, Latest, Objective.

adding new knowledge to a "heap". Knowing the "intent" is also important to resolve some ambiguities of interpretation as will be explained below.

7. Information-Exchange Session Types

Information is exchanged through *messages.* Messages sent may invoke responses, which may lead to further messages – in many different patterns among two or more parties. A set of such related messages is here called an "information exchange session". Each session has a particular purpose. There are many different purposes for exchanging information and these may be categorised into "information-exchange session types". Such a categorisation is useful because it enables us to tailor the proposed common language to distinguish between them, and this will help machines to extract and interpret information from messages accurately – because the pragmatic part of the meaning of the message is implicit in the role of the message within an information-exchange session of a given type.

So, using the example of section 4 again, in our common language we would explicitly declare in the first messages of a given information-exchange session that the session is a "propose-approval" session, or a "command-acknowledge" session or an "inform" session. This would then resolve the semantic ambiguity of what "yes" means in the above example (approval, commitment, or acknowledgement respectively), and if each information-exchange session type is defined in a machine understandable form the machine will know what it should do as a consequence of receiving a given message (i.e. the pragmatics can be defined for each message within a given information-exchange session type).

The list of VKB Information-Exchange Session Types is still being developed but includes:

- Inform (the passing of new known-facts for addition to a knowledge-base)
- Show (the opposite of inform: the export of information derived from the VKB's internal representation of *known-'facts'*).
- Question – Answer (User posing a question and the VKB providing a response from the known-facts that it contains)
- Mediated Information Exchange – this is a class of Information-Exchange wherein a User wishes to communicate some information to one or more other users via the VKB and the VKB provides some value-added service in the process. This value added could be: recording of exchanges – including acknowledgements, approvals, authorisations etc. for legal purposes and for future reference by users; to keep track of the status of a sequence of information exchanges such that users can query their status; to keep track of the impact of the information exchanges on real-world objects through an internal model (e.g. modelling of resource commitments made through information exchanges); handling the delivery of information to the specified or appropriate users, including and appropriate security handling measures; etc.. The following are examples of Information-Exchanges that fall into this class:
    - Propose - approve/reject/counter-propose
    - Command –Acknowledge (with implied commitment)/ Clarify
    - Request – Response
    - Transaction (an atomic commitment to a set of changes, not necessarily related to the exchange of goods for money)
    - Vote
    - Auction
    - Poll

- VKB Directive (a direction - request or command - from a user/administrator to be passed to a VKB service to change the configuration of the VKB and/or the services it is providing; with optional response).
- Notify (an output of a VKB notifying Warfighter Information Services subscribing to this service that a event has occurred for which notification was requested – an event here meaning the known-facts changing in some defined way – this could entail complex-event processing).
- Synchronisation (ensuring that different instances of a VKB, or external partners, share some set of information-objects or known-'facts').

The common language proposed here will support all the above information-exchange types.

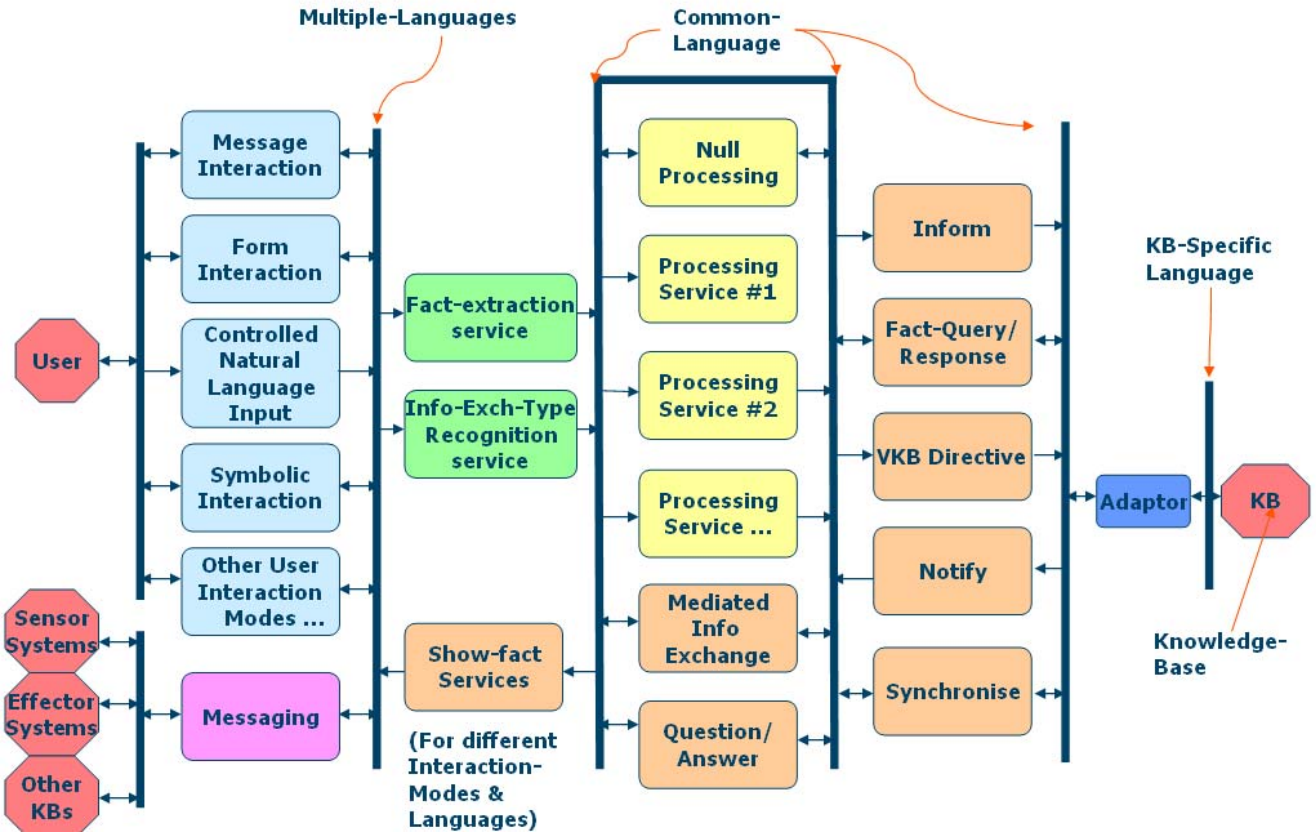

Figure 5 - Languages used at different interaction interfaces

Figure 5 illustrates which interfaces will use the proposed common language, and which will use other languages. (The inverted U shaped bus in the centre connects outputs back to their inputs; so, for example, the Mediated-Info-Exchange service could call a Processing Service, before passing the result to the Inform Service and thence to a Knowledge-Base.) The interface to each Knowledge-Base is specific to the technology used to implement the knowledge-base. An adaptor is used to translate between the common-language and the KB-specific language. For example a KB could be implemented as a Prolog proposition-store and have a Prolog interface; or it could be implemented as a data-base and have a SPARQL query interface and other interfaces for data-entry etc.

(The above diagram does not show the control logic necessary to orchestrate the services. Which services call which other services and pass what information/facts to them under what conditions would be defined through business rules acting on the facts presented at a series of decision-points.)

8. Outline of a Common-Language for Warfighter Information Services

The common-language will have two components: propositions; and ways of representing the pragmatics of an information exchange. Both components will need to be serialised - XML being the most common means.

As explained in the previous section the pragmatic component is dependent on the type of Information-Exchange Session to which a given message belongs and the role of a message within a session of exchanges. The pragmatic component will also need to contain session-IDs/message-IDs and the like used to correlate related messages. The pragmatic component will contain the information necessary to place the propositions in the context of a known pattern of Information-Exchanges and hence indicate what needs to be done with them. The details of what should be done with each proposition are not contained in the messages: they are contained in externally-supplied business rules. The common language must provide the vocabulary and semantics which is shared between the pragmatic component of the messages and the external business rules. The key point is that what are stored in the knowledge-base are sets of propositions, not messages[4]. A purpose of the pragmatic component of messages is to help decide how propositions should be assimilated into a knowledge-base, i.e. how the new knowledge affects what was previously known – as propositions - and to clarify which qualifier-flags should be applied (see section 6). Assimilation of propositions is discussed further in section 13.

Information-exchange messages will not usually contain propositions that are fully qualified. This is because the qualifications are inferred from the "context" of the propositions, i.e. from the pragmatic component of the common-language which is defined for each Information Exchange Session Type.

For example, in a "request" information-exchange session, it is implicit that the thing requested (a proposition or set of propositions) is at first "affirmed (true)", "hypothetically", "categorically", and in an "Alternative World" (requests are not statements about what is so now in the real-world). If the request is approved the "thing requested" would be qualified as "affirmed (true)", "declared", "categorical", and "Alternative World". Subsequently if the thing requested and approved is then physically enacted (e.g. a purchase), the same set of propositions describing "the thing" would be qualified as being so in the "Objective World".

The pragmatic component also has a second purpose which is to help determine what action is needed to create a response – and to determine whether or not a response is required.

This paper fully defines an abstract representation of propositions and their qualifiers, but does not provide the language for the pragmatic component of each information-exchange session-type. This is the next research task. It is expected that suitable languages will already exist for some of the session-types but not others.

9. Semantics of propositions

The sense of a proposition is the fact(s) that it represents. The truth of a proposition is determined by whether or not the proposition corresponds with reality[5]. In the knowledge-bases and common-

[4] Messages may also be stored separately and cross-referenced from the knowledge-base.

[5] The first two sentences of this paragraph follows from Wittgenstein [1]:
2.221 What a picture represents is its sense.
2.222 The agreement or disagreement of its sense with reality constitutes its truth or falsity.
2.181 A picture whose pictorial form is logical form is called a logical picture

language proposed by this paper, propositions represent facts through a *conceptual-model* of entities, attributes and relations, which are expressed as propositional statements using the abstract representation standard defined at Appendix A. This standard also assigns every proposition to one of several different kinds of "set". Sets, and operators acting on them, provide the means to combine propositions into more complex statements about which machines can reason. All this depends on having an appropriate conceptual-model.

10. Building the conceptual-model

Propositions are made using tokens (labels) that stand for ideas that exist in a conceptual-model[6]. The conceptual-model is an ontology – i.e. it defines what is possible to exist. Propositions make assertions/declarations that what is possible to exist does in fact exist – or not – in the world.

It is important to emphasise that the activity of ontology creation must be focussed on the purpose for which the known-'facts' are to be determined – e.g. to support Situational Awareness and Decision-Making. The question to be answered is: "what do we need to know to make good decisions?" not: "how can I model everything that is so in the world?"

In this paper it is proposed to partition the Virtual Knowledge Base into representations of four "worlds": the Objective-World, the Subjective-world, the Alternative-worlds and the Universal-world.

The Objective and Subjective worlds both refer to the real-world. The difference between these is the difference between "fact" and "opinion", which can be distinguished more specifically by considering the perspective from which the proposition is being made. If what is being stated is asserted or declared to be so (or not so) in the real-world is independent of the observer then it is classed as Objective. If however what is important about the statement is that it is the view of an individual, or group which exists in their mind independently of objective facts then it is qualified as being a statement in the Subjective World. (A fuller explanation appears at Appendix A.)

The Alternative-world refers to own-plans and external-world conjectures about which there is currently no commitment or consensus belief in existence. The Universal-world holds knowledge about the world that is universally true in all time-frames and all contexts, e.g. knowledge of possible, contingent and impossible existence (i.e. ontologies).

Each of these worlds is considered to be populated with the same kinds of *Entity*, which are either physical or abstract[7]. Each entity has a number of attributes and values of those attributes, and the each entity will have relationships with other entities. *Ontologies* shall be used to define classes of entity that are of interest the worlds, and the possible (relevant) attributes, values and relationships. Knowledge about these worlds shall then be represented in the VKB by identifying entities in each world as instances of the entity-classes in the ontology, and recording their attributes, values and relations in accordance with what is permitted by the ontology – and expressed as propositions in the logical structure defined at Appendix A. The Universal World will also contain Axioms.

---

3. A logical picture of facts is a thought
3.1 In a proposition a thought finds an expression that can be perceived by the senses.

[6] Words as tokens for ideas – as per Locke [2]
[7] Examples of abstract entities are tasks and events.

11. Axioms define the meaning of relations

If we wish Warfighter Information Services to act on propositions it is not sufficient that propositions represent reality: the services also need to know how to reason about them, which requires an understanding of the entailment or consequences associated with something (represented as) being so or changing from one state to another. This extra knowledge needs to be provided through a set of axioms. In effect, axioms define the "machine-meaning" of relations[8] between entity-classes, where the term "machine-meaning" here refers specifically to the consequences for a knowledge-base of these relations existing between entities represented in a knowledge-base. (Of course these axioms should be constructed such that they are "truthful"; i.e. that what they determine is so in among representations in a knowledge-base is also true in all cases among the objects in the real-world that the knowledge-base is representing.)

Examples of axioms are:

- If entity-class A "is-a" (subset of) entity-class B, then what is predicated on entity-class B is inherited by all the instances of entity-class A.
- If A has relation X to B; then B has relation Y to A (e.g. X=father-of; Y=son-of).
- If A is a "part-of" B; if B ceases to exist then all its parts cease to exist.
- If A is "targeting" B and A "is-a" instance of entity-class "missile"; then A "is-approaching" B and A "intends-to-strike" B. (This example is taken from reference [15] which is very instructive on how symbols acquire meaning.)

(Where "is-a", "part-of", "targeting", "intends-to-strike" etc. are all relations.)

12. Fact-Extraction from textual inputs

The foregoing discussion has been concerned with how expressions can be formed in a common-language, incorporating propositions and XML tags, for use by Warfighter Information Services and acting on propositions in a knowledge-base. This section considers the next problem which is to how to process external textual inputs from users/systems to extract *'facts'* that a community wishes to know[9] (N.B. not all facts). These *'known-facts'* are recorded as propositions in the format of Appendix A. Such propositions record that:

- An instance of a certain entity-class is said to have a given attribute-value predicated on it.
- An instance of a certain entity-class is said to have a given relation to another instance of the same or different (usually different) entity-class.
- Plus various qualifications, distinguishing whether the fact is concerning the present, past or future, and whether it has been asserted or declared, etc.
- Associated information about a proposition (or set thereof): e.g. provenance, references, confidence, accuracy, etc., recorded as 2nd-order propositions.

It is important to realise that extracting *facts* is different from extracting *words*. What are being extracted here are the concepts that the words stand for, and the logical relations between these, which is independent of the language in which the words are expressed. Actually *words* is too narrow a term for what is being performed here, the correct term is *token*: - a token may be a word, abbreviation, acronym, alpha-numeric sequence (e.g. date or part-number) etc..

[8] Relations equate to predicates and verbs in other authors' descriptions of this subject matter.
[9] Communities specify what they wish to know through their own conceptual-model express as an ontology.

To do this requires a conceptual-model, expressed as an ontology, which defines a set of concept-labels (as entity-classes, attributes and relations, and possible combinations thereof). A data-dictionary and extraction rules associated with the ontology are also required to define how to map from the *tokens* to the concept-labels.

Unfortunately it may not always be possible to constrain the conceptual-models that those submitting information to a VKB will use, so it will often be necessary to support a set of conceptual-models for inputs from different sources. Secondly, and independently of the previous point, the conceptual-model(s) for the inputs may or may not be the same as the conceptual-model that a particular community creates to record what they want to know. If for either of these two reasons the conceptual-model for a particular information-object being processed is different from the community's own conceptual-model then a two-stage fact-extraction process will be needed:

a) To extract facts using the conceptual-model of the input text.
b) To infer from these extracted facts what it was that the community wished to know, and to record these new facts in the VKB (discarding the results of step a). This inference process is here called schema-mediation.

The community's chosen conceptual-model for representing what it wants to know may simply be a subset of a larger conceptual-model used for information input, in which case the mediation is simply a question of filtering and discarding what the community does not wish to know; or the community's conceptual-model may be on a completely different basis and hence require more sophisticated processing to achieve the required mediation.

As the VKB will need to support conceptual-models devised by multiple communities and support the conceptual-models used in many different kinds of information input, when a concept is extracted its label must be prefixed by a unique conceptual-model identifier. This is similar to the existing practice in XML to use name-spaces, except in that case the prefixes only have local significance (in a given document) and so do not need to be unique. Here all the facts extracted from many documents are being put into the same "pot" so unique prefixes are needed within a given knowledge-base.

The first step of fact-extraction is to analyse the *tokens* to determine which *concepts* they are instances of within the conceptual model provided by a schema of entities, attributes and relations. For example that the series of numbers 12-10-09 is an instance of the concept of a Date, or that John Brown is an instance of the concept of the FullName-attribute of an instance of the Person entity-class. Note that the *word* "date" appearing in the input text as a token, is quite different from the *concept* Date, which appears in the input text in this example through in the form of a string of numbers (12-10-09). This first step is the same as Named-Entity Recognition, and existing tools for this could be incorporated into the service. However commercial tools mostly focus on the nouns, noun-phrases and dates/times etc. What is required here is also the recognition of verbs, and the recognition of 'parts of speech' (prepositions, articles, pronouns, connectives …).

At this point in the process there will often be more than one possible interpretation of the tokens in the information-object (a 'tank' might be a fighting vehicle or a vessel for holding liquids). All options should be taken forward at this stage and resolved later (if possible). If it is not possible to resolve ambiguities then a set of mutually exclusive propositions should be created (i.e. a set about which it is known that only one member is valid – see Appendix A).

How hard the process is of mapping from input tokens to concepts depends on whether the input consists of:

- Natural Language (very hard) - unconstrained vocabulary and arbitrarily complex syntax (including inputs that do not conform to the rules of the Language but are still meaningful to humans); morphology has to be addressed (tense, active-passive, plurals, subject-verb agreement for regular and irregular verbs).
- Controlled Natural Language (quite hard) - constrained vocabulary; mapping of vocabulary to concepts has to be defined through a data-dictionary, simpler syntax than Natural Language but still quite hard, morphology has to be addressed.
- Mark-up language (e.g. XML and its dialects) (easy) - Tags define the concept that applies to the token that they surround: <concept-label> token </concept-label>; constrained vocabulary of tags; simple syntax; no morphology.
- Formatted-messages (easy) - Through the associated schema, the position of token in the input sequence determines the concept that it is an instance of; fixed syntax of each message-type; no morphology.

The next step is to use the syntax of the language[10] (and where present, the morphology) to determine what is being predicated on which subject. Additional rules may also be required to define more complex ways of extracting the correct representation from the input text. This amounts to explicitly defining the semantics through an extraction rule. An example of such an extraction rule is shown in the box below:

Example of an Extraction Rule:

If the input text says that "A killed B" and if A and B are names of people then assert the following 4 propositions:

entity-class ="Person", instanceID=xyz, attribute= "live_or_dead"; value="dead"

entity-class ="Person", instanceID=xyz, attribute= "name"; value="B"

entity-class ="Person", instanceID=xyz, relation= "killed_by"; entity-class=Person, instanceID= abc

entity-class ="Person", instanceID=abc, attribute= "name"; value="A"

The box above shows how machines "understand" the text – they derive the meaning from the semantics, syntax and morphology of the text, as defined by rules. In this example the rule (implicitly) defines:

- The vocabulary definition of the verb "to kill" as being - to render the live_dead attribute of an instance of the "person" entity-class to have the value "dead".
- That the syntax of the language is that to kill is a transitive verb, that the word order and absence of the preposition "by" indicate the active not passive and so the actor here is A, and the object is B (and so the instance of person that has the attribute live_or_dead set to "dead" is B not A, and similarly that "killed_by" points to A not B)

[10] In natural languages (and controlled natural languages) 'parts of speech' (prepositions, pronouns, articles, and connectives), punctuation and word order are the main components of the syntax. Parts of speech are not normally used mark-up languages and formatted messages, but the token order and punctuation are usually very significant.

- That the morphology of the language is that killed is the past participle of the verb "to kill", and so appearing in the definite past it is appropriate to make the assertion that B's live_dead attribute is now "dead", whereas if the morphology indicated the future tense then this would not be appropriate.

Ambiguities in the initial mapping of tokens to concepts may be resolved at this point by reference to the relevant conceptual-model (ontology). For example it would become evident that 'tank' is either a class of vehicle or a class of vessel because the syntax will show which attributes belong to this subject, and the ontology will show that some of these attributes are possible for one interpretation of the token but not the others[11]. Ambiguities in attribute and relation tokens can be resolved in a similar manner.

Some tokens may not be recognised. This can be for two reasons:

- Because they are not relevant to what the community responsible for the extraction wants to know, which it specifies through its schema for knowledge representation.

- Because the tokens are not in the vocabulary over which the schema's extraction rules have been defined.

It will always be hard to distinguish between these two cases. The first is an intentional decision to ignore certain subject matters. The second is a deficiency in the extraction design process. Several strategies for handing this dilemma are possible; some erring on the side of caution and some the reverse - ignoring unrecognised tokens. Cautious approaches include prompting humans for assistance, recording unrecognised tokens for off-line human processing (perhaps sorting by frequency of occurrence first), using web-searches, dictionaries and thesauri to "expand" the unknown token in the hope that the expanded term will match a known concept.

The fact-extraction service will initially assign a new unique instance-ID to every entity in the input text, except where the syntax indicates that a set of statements are referring to the same entity. This service has no way of knowing whether the entities in this piece of text are the same entities referred to in other texts, or are known about from other form of input. Determining which entity-instance-IDs are referring to the same entity in the external world is a separate activity, which is performed by a correlation service.

When defining fact-extraction rules it is often not necessary to proceed via the two steps of:
(1) literal meaning extraction (from general purpose dictionaries, syntax, pragmatic indicators etc,)
(2) significance determination

Fact-extraction can be achieved in one step by encoding the significance into a "dictionary" that is specific to a community. The "community-specific dictionary" defines what some text "means for them" in their knowledge-representation schema which is not necessarily what it literally means. For example: an intelligence community may be interested in knowing about the "known-associates" of certain people, which they may choose to record by means of a relation "is-associate-of" between two entities of class "person". They may receive a report saying that "person A was seen *talking to* person B, at a certain place and time." What this community is interesting in knowing from this report is that

[11] This method of disambiguation works best if the schema also defines what is impossible. At first sight this may seem intractable (there being an infinite number of impossible things), but some very broad statements can be made (as an upper ontology of what is impossible) – e.g. that inanimate objects cannot have verbs of volition predicated on them, or that immaterial entities cannot have material attributes etc.

they can now add a new piece of knowledge to their collective knowledge base that "Person A is an-associate-of Person B". They can do this in one step by defining "talking-to" to mean "is-an-associate-of" by having a knowledge representation dictionary entry of "is-associate-of" with "surface forms" (literal text) of "talking to", "met with", "exchanged money with" etc., none of which literally mean "is-associate-of" but all of which can be taken to imply (signify) this for the purposes that this community is interested in. This being simpler than the alternative of first extracting the fact that "(subject)A , (relation) spoke to, (object) B, and then defining an inference rule to say that the relation "spoke to" implies the relation "is-associate-of" and hence to infer that "(subject)A, (relation) is-associate-of, (object) B".

13. Assimilation of newly extracted known-facts into a Knowledge-Base

When new known-facts have been extracted they need to be correlated with what was previously known to determine whether or not they refer to the same entity-class instances. If such a correlation is established then the entity-instance-ID of the new facts will be changed to match that of the previously known facts. But there is more to assimilation of new-facts into a knowledge-base than correlation – these extra activities are what the "*Inform Service*" does as described below.

The *Inform Service* will distinguish facts that are *asserted* and *declared*, and check the authority, authenticity and integrity of declarations (as and if required by supplied business rules).

If a new 'known-fact' is marked as being "the latest"[12] then the service will query the previously known facts to determine what was previously asserted or declared to be "the latest". If such a fact exists, its validity time (recorded as a 2nd-Order proposition) will be compared with the new fact and the latest one determined. The latest one will then be marked as such, and the older one marked as "historic".

The Inform Service will check whether or not a new fact that is stated Categorically contradicts previously known Categorical facts, and if it does then the set of contradictory propositions will be identified as a mutually exclusive set. If a new fact is stated probabilistically, the Inform Service will query what is already known to find other propositions stating the same fact. If such propositions are found and they are also stated probabilistically then the set of propositions will be identified as a mutually exclusive set. If some facts are stated categorically the same facts are also stated probabilistically then these are not identified as contradictions. When searching for contradictions assertions are compared with assertions and declarations with declarations – not mixing assertions and declarations.

The Inform Service will check to see if new facts resolve ambiguities that were previously noted (and recorded as mutually exclusive sets of propositions – see Appendix A). If the new fact is asserted as true that belongs to such a mutually exclusive set then fact previously recorded as possible will be set to "true" and the other possible alternatives set to "false". If the new fact is "false" then the action is simply to discount this option by setting the previously recorded fact to "false" instead of possible.

14. Other Knowledge-Base Interaction Services

Some of the other services of Figure 5 are explained below, for which language is a key issue.

[12] Every "known-fact" (a.k.a proposition) is classified as historic, latest, future – see Appendix A.

The *Show Service* will articulate selected 'known-facts' as textual messages, or as completed forms, HTML web-pages, geographic-overlays, symbols, text for avatars to speak etc. Inputs to the service specify which sets of facts are to be articulated, and the modality/language of presentation to be used. This is the opposite of fact-extraction: how what is known in a knowledge-base is communicated to users in a form that is easy for them to understand.

The *Query-Service* will determine if a set of stated propositions "matches" what is known, across the whole distributed knowledge-base. If the stated propositions contain variable names then the service will identify known facts that match the input parameters that were not variable and then return the values of the variable parameters from these matching facts (or a fail response). If the input propositions did not contain any variable names then the response will be True or Fail (fail meaning 'not known'). This service needs to operate in an efficient manner across a distributed knowledge-base, and this will entail the service itself being distributed with the processing taking place at the locations that will minimise loading on the network to determine the results.

The *Mediated Information Exchange Service* will mediate Requests, Commands, Proposals, Approvals/ Authorisations, Transactions, Offers/Bids, Polls (and perhaps other types of Information Exchange) between external parties via a knowledge-base. Mediation may entail determining where to send requests or commands, and how (which communications channel, with what QoS/Information Assurance), and in what form/format (converting as necessary), keeping auditable records of messages sent and received (to support non-repudiation), etc.. It may entail brokering (matching offers to bids), and the checking of authenticity and integrity. Mediation also entails keeping track of the state of patterns of message-exchange (e.g. request-made, awaiting response … accepted/counter-proposal made …), and associating each new message with the correct set of exchanges. Inputs to the Mediated Info Exchange service will be sets of facts extracted by the Fact-Extraction and label the *type* of Information Exchange. Outputs will be sets of facts which are then articulated as an Information-Object via the Show Service, and delivered by the messaging service (with signature/encryption if required).

15. Summary

In the Virtual Knowledge Base concept being proposed here the intention is that a knowledge-base will store "known-facts" and that, in principle, any service should be able to access to reason over everything that is known in a given knowledge-base. To achieve this, the semantics of the known-facts have to be "understandable" by all the services. The W3C's semantic-web initiative addresses this need but its focus has hitherto been on making fairly simple statements about "resources" that exist. It has been shown that it is not sufficient to make statements consisting only of a subject, predicate and object: it is also necessary to "qualify" each statement to make a set of important distinctions between statements having the same subject. This is because in military knowledge-bases we wish to store statements about the past, present and future, distinguish between assertions and authoritative declarations, to record claims that that something does not exist or is not so, or may be so (and allow contradictory statements to be recorded), be able to handle ambiguity, inaccuracy and probabilistic uncertainty, and make statements planned or hypothetical situations without confusing these statements with statements about the world as it really is. (And distinguish a hypothesis about the current situation from a hypothetical situation which is known not to currently exist, or at least in which there is no entailed implication that it is believed to currently exist.) It has been shown that all these important things can be achieved through the use of a set of defined "qualifier flags", supplemented by "2nd-order propositions" (propositions about propositions). A proposed sufficient set of "qualifier flags" has been fully specified (at Appendix A), and a list of different kinds of 2nd-Order propositions has been proposed, though this aspect needs further details to be added. It is expected that

world-wide meta-data standards will be re-usable here, but some additional new standardisation work is also likely to be needed.

It has been shown that Humans will need to interact with Knowledge-Bases for different purposes, which can be characterised by defining different Information-Exchange Types (e.g. Inform, Question, Mediated Information Exchange between users via a VKB, Directive to VKB). Human interactions can be presented in many languages, of different kinds: natural languages, controlled natural languages, and machine-orientated (e.g. RDF/XML). It is not sensible to expect every service in a VKB to "understand" every language that a human might use to encode his/her interactions. It is more efficient to use a common-language within a VKB and then to translate into human-accessible form at the interface between the user and the VKB. A common-language is also necessary to achieve "information fusion" between inputs presented in different languages, i.e. the knowledge-base should hold its known-facts in a manner that is independent of the user-interaction language, here presented as a formal language of propositions. Here the task of translation between the human-interaction language and the common-language is performed by two services: fact-extraction and show-fact, for input and output translation respectively. It has been explained how the semantics of a human textual input (of any kind – natural/machine-orientated) can be extracted into a set of propositions, through rules operating on the vocabulary, syntax, and where present, the morphology of the text - all on the basis of a conceptual-model of entities, attributes and relations defined in an ontology. But human-interactions do not just contain propositions: the propositions are presented in the context of some intentional activity, and this needs to be understood as well as the literal meaning of a message; i.e. we need to understand the pragmatics of a message as well as its semantics. It is proposed to capture the pragmatics by classifying sets of messages-exchanges into a set of Information-Exchange Session Types, and then to provide a service dedicated to extracting information about the exchange type and about the role of a given message within the pattern of message-exchanges that a given exchange-type comprises. This information can then be used to determine what a given word or phrase means in this context, and what should be done with the extracted facts (with the aid of additional rules). In the proposed common-language (used within the VKB) this extracted "contextual" information about a message will be serialised along with the extracted propositions, using a suitable serialisation standard – e.g. XML.

But a VKB's knowledge-base of "known-facts" is not the set of input messages, nor these messages translated into the common-language. It contains only propositions about the way a world is, was, will, or might be. The propositions contained in the input messages translated into the common-language are not simply "added to the heap" in the knowledge-base. They are assimilated (via the Inform Service) in a way that considers how what it has just been told affects what it already knew, and taking into account the "intent" of the message – defined through rules acting on the extracted contextual information as discussed above. Having determined all this new known-facts are added (and/or existing facts modified) to the knowledge-base, and the message itself is put to one-side – in a separate information-object store, as a record for reference purposes. (Normally a reference to this record will be included as a 2^{nd}-order proposition asserted about the set of 1^{st}-order known-facts that were extracted, or inferred, from it. This enables humans to review the provenance of the known-facts.)

Other services within a VKB, not described here, will use the set of known-facts in a VKB to help military staff to attain situational awareness and make plans and decisions. For this to be effective it is vital that the semantics of the contents of a VKB (its propositions) are very clearly "understood" by the services and the humans. (There could be quite dangerous consequences if this were not the case.) This paper has attempted to establish a firm basis for the semantics of propositions in a VKB by defining them in a formal language (Appendix A). This has been done by means of a directed graph representation which is fully compatible with RDF. In any given implementation a choice would have

to be made of a standard for serialisation of this graph. This could be XML, but it could also be KIF, Common-Logic, or yet other means.

It is a consequence of introducing 2nd-order propositions, and qualifier-flags that the reasoning processes associated with the propositions will be more complicated. This is clearly not desirable – but it is here claimed that this is necessary in order to build knowledge-bases that are fit for military use, given the complexity of real-life situations, with all their ambiguities, uncertainties, probabilities, hypotheses etc. – not to mention the need to reason about possible deceit!

16. Future Work

The next stage of the work is to complete the definition of the "common-language" including its pragmatic components for each Information-Exchange Session Type, and selection or development of a standard vocabulary for the predicates in 2nd-order propositions. Then these ideas will be validated through a set of Use-Cases, using the Warfighter Information Services identified here and a prototype knowledge-base. Early work will focus on fact-extraction from XML-tagged forms, and ADatP-3 formatted messages and then move on to the use of Controlled Natural Languages.

References

[1] Lugwig Wittgenstein, "Tractatus Logico-Philosophicus", 1922. www.gutenberg.org/etext/5740

[2] John Locke, "An Essay Concerning Human Understanding", 1690. www.gutenberg.org/etext/10616

[3] Immanuel Kant, "A Critique of Pure Reason", 1781. www.gutenberg.org/etext/4280

[4] John R. Searle, "Expression and Meaning: Studies in the Theory of Speech Acts", 1979, ISBN 0 521 31393 7

[5] "A Wittgenstein Approach to the learning of OO-Modelling", C Holomboe, Computer Science Education 2004, Vol 14, No 4, pp277-296.

[6] Dr Dale Lambert, "A blue-print for high-level information fusion for situational awareness", Information Fusion 2009, Vol 10, Issue 1, pp6-24.

[7] Peter Roget, "Thesaurus of English Words and Phrases", 1852, www.gutenberg.org/etext/10681

[8] "Knowledge Interchange Format" (KIF). www.ksl.standford.edu/knoweldge-sharing/kif/

[9] "Information Technology – Common Logic: a framework for a family of logic-based languages". International Standard ISO/IEC 24707. First edition 2007-10-01.

[10] W3C, "Resource Description Framework", 2004-02-10, www.w3c.org/standards/techs/rdf

[11] W3C, "RDF-Semantics", Feb 2004, www.w3.org/TR/rdf-mt

[12] W3C, "RDF Vocabulary Description Language 1.0: RDF Schema", Feb 2004, www.w3.org/TR/rdf-schema/

[13] Ludwig Wittgenstein, "Philosophical Investigations", Wiley-Blackwell, Revised 4th Edn 2009, ISBN 978-1-4051-5928-9.

[14] W3C, "Ontology Web Language Reference", 2004-02-10, www.w3c.org/TR/owl-ref

[15] Dr Dale Lambert, Dr Chris Nowak, "The Semantic Challenge of Situation Asssessments", 2005 7th International Conference on Information Fusion.

Appendix A: Propositions and Sets in a VKB

This appendix provides a full specification of a proposed abstract representation of *propositions*. Each proposition is assigned to at least one *set*. This appendix defines different kinds of set and shows how they can be used to form complex expressions from multiple propositions – in a logical way that supports machine reasoning. The chosen form to specify the abstract representation is a directed graph. This permits wide application of the ideas in different languages including the semantic-web's set of standards (OWL/RDF/XML etc.), Common Logic or KIF. It also permits different standards for serialisation of the graph to be employed – yet in all cases ensuring that the results are isomorphic, such that there is equivalence of meaning whichever method is chosen.

A1. Propositions

Propositions are used to represent everything that we wish to state that is, was, is expected to be, may be, or not, in the real-world or in an alternative-world. (An example of an "alternative-world" is a plan. The entities in a plan may be identical to entities in the world-as-it-is, but propositions about these two contexts clearly need to be distinguished.) In addition propositions can record what is possible to exist, and relations among what is possible to exist (in the real-world or alternative-worlds). These propositions form an ontology, and in this scheme they are distinguished from statements about the real-world.

Statements have exactly one subject, predicate and object (but subjects and objects can be sets – see A3).

When making statements about the real-world, or an alternative world, the subject of a statement must be an instance of a defined class. Classes are defined through an associated ontology – which can be stored within the schema defined here, in the Universal World, or can be defined externally by other means, e.g. in OWL. Objects can be instances of these classes or literals.

A fully qualified proposition consists of a statement and a set of qualifier flags as indicated in Figure A-1.

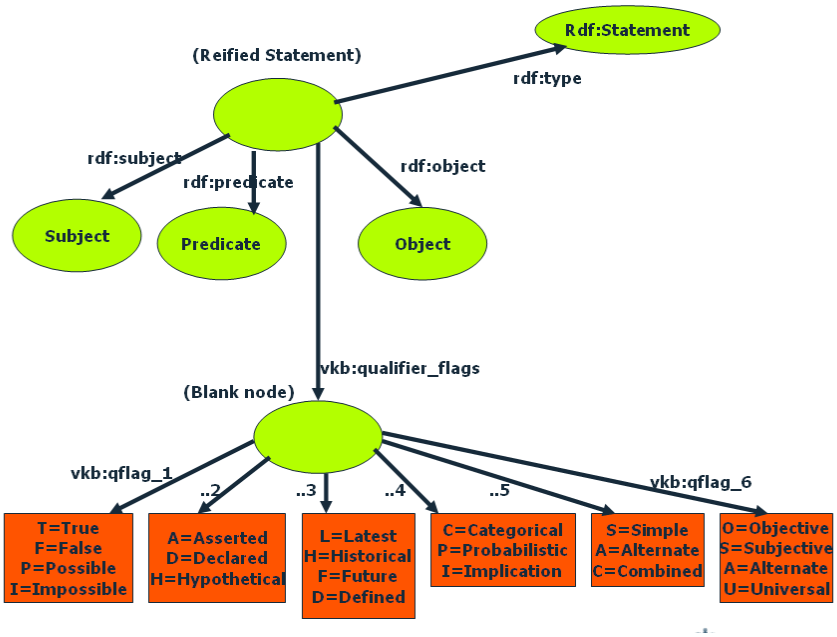

Figure A-1: Fully qualified proposition

The allow values of the qualifier flags, and their meaning is defined below.

vkb:qflag_1

The allowed values of vkb:qflag_1 are “T, F, P, I” (True, False, Possible, Impossible) for statements about the Objective, Subjective or Alternative Worlds and “N, C, or I” (Necessary, Contingent, or Impossible) for statements about the Universal World.

The values “True” and “False” are used to distinguish between propositions that assert something is so (T), from propositions that assert something is not so(F). (Whether or not the whole proposition (including this truth flag) is actually *‘true’* depends on whether or not the proposition corresponds with reality in the specified world.) As a design principle of the VKB it has been decided that the *open world principle* will apply. This is in contradistinction to the *closed world principle* (which is often used in knowledge based systems) in which we only have propositions about “what is” and the absence of such a proposition indicates that it is not so or does not exist. In our VKB what is, and is not, are explicitly stated, and the absence of either statement is interpreted as unknown. Note also that this form of representations also permits contradictions to be held in the knowledge base (that something is both true and false in different propositions). The ability to handle contradictions, i.e. to reason about which ‘fact’ is correct, and know whether ‘facts’ are corroborated or not-corroborated by independent evidence sources is a vital part of military intelligence, and it is part of establishing the appropriate level of trust in any information or intelligence in the VKB, which is important because there will be life-and-death consequences arsing from decisions made using this information/intelligence.

“Possible” and “Impossible” are other valid values of vkb:qflag_1. If these values are used (and it might be that they are not, or just possible and not impossible) then the logic is extended to be 3-value logic, or 4-value logic, respectively. *Possibility* is distinct from *probability*: there may or may not be a probability associated with a possibility (usually not) – this is indicated by vkb:qflag_4. The main use envisaged for the Truth value of “Possible” is to cope with ambiguous inputs, which are captured as mutually exclusive Alternatives, as explained below in connection with vkb:qflag_5.

vkb:qflag_2

The allowed values of vkb:qflag_2 are “A, D or H” (Asserted, Declared Authoritatively or Hypothetical). Anyone (or system) may make an assertion, from their own observations or opinions. Authoritative declarations can be made only by those (including automated systems) authorised to make them. For example this flag has may be used to distinguish “reports/observations” from “assessments”, and to distinguish individual object observations from a correlated/fused view which is declared as being authoritative (e.g. the Recognised Air Picture). Hypothetical statements can be made singly or as mutually exclusive alternative hypotheses. In both cases hypothetical statements hypothesise that something is, was, or will be so, or not so. They are thus ascribed a vkb:qflag_1 value of “T” (true) or “F” (false) (with the timeframe indicated by vkb:qflag_3). These hypotheses may be about the current situation, in which case vkb:qflag_6 will be set to refer to the “Objective” or “Subjective” World; or they may be about a planned or hypothetical situation, in which case vkb:qflag_6 will be set to refer to an “Alternative” World.

vkb:qflag_3

The allowed values of vkb:qflag_3 are “H, L, P, D” (Historical, Latest, Future, Defined). The “Defined” value is used to qualify statements that are independent of time. The other values refer to the timeframe of the statement they are qualifying relative to the time at which the qualified proposition was created. The passage of time after the proposition is recorded does not change how it is recorded (i.e. a statement about the future will eventually refer to events in the past but they are still recorded as “future-statements”: they must be distinguished from statements that are observations about what actually happened.) The meaning of a future statement depends on the world it is referring to - as defined by vkb:qflag_6. In the Objective/Subjective Worlds future-statements are either predictions or expectations. In the Alternative world future-statements are intentions or assumptions in the context of a plan or hypothetical situation. In this scheme, statements about what “is now” (in the real-world, Objective or Subjective) are initially qualified as being “Latest”. However, when assimilating new propositions claiming to be “the Latest” into a knowledge-base the veracity of this claim needs to be checked such that there is only one “Latest” proposition in a knowledge-base referring to some time-varying attribute or relation of a given entity. (Determining which proposition is the latest will be done by referring to a 2nd-Order proposition which defines the date-time at which the original proposition was claimed to be valid.) Often, during this assimilation process, a previously “Latest” proposition about the same entity-attribute/relation will be found. In this case the qualifier on this older proposition should be changed to from “Latest” to “Historical”. It is also possible to make propositions about the past, which are known not-to-be-so now. These are qualified as “Historical” from the outset.

vkb:qflag_4

The allowed values of vkb:qflag_4 are “C, P, I” (Categorical, Probabilistic, Implication). Categorical (1st-Order) propositions are simply stated to be true, false, possible or impossible. They do not have an associated 2nd-Order proposition that provides a probability that the referenced 1st-Order proposition is so, or any associated . Whereas if this flag is set to “P” this means that there is a 2nd-Order proposition provided that contains the probability that this 1st-Order proposition is so. Implication statements assert, declare or define (qflag_3) a logical dependency between the truth of a given statement and the truth of another statement or set of combined statements. If qflag_1 on the implication statement is set to T (=True) then this statement is true IF the other statement, or combined set of statements, is/are True. If qflag_1 on the implication statement is set to F (=False) then this statement is true IF the other statement, or combined set of statements, is/are False. A 2nd-Order proposition is used to link this statement to its antecedent(s). Implications can be recorded in any World (vkb:qflag_6), including Universally defined implications for whole classes of entity.

vkb:qflag_5

The allowed values of vkb:qflag_5 are “S, A, or C” (Simple, Alternate, or Combined). This flag indicates whether the proposition is simply stated as being so (Simple), or that there are a set of mutually exclusive (Alternative) propositions only one of which can be so, or that there are set of propositions which are to be logically combined (in a manner specified as 2nd-Order proposition linked to the set). Such logical combinations include AND, OR, XOR, but also A if B is so, or A if not-B etc. *Alternate* propositions may be used to record ambiguity and alternative hypotheses - which are handled differently as follows. To handle ambiguity, the Truth value of each Alternate proposition would be set to “Possible”. Then if the ambiguity were ever resolved, categorically, then the original proposition’s Truth value would be changed to “True” and all the other Alternative propositions’ Truth values set to “False” (because they are mutually exclusive). If the ambiguity was resolved but in a

probabilistic manner and not in a categorical manner another proposition could be added (a Simple one) with Truth value “True” and a probability assigned (as explained below), leaving the all the Alternate propositions as they were (all Truth=possible). In the case of hypotheses, each alternative hypothesis would be recorded as member of a set of Alternative propositions, their Truth value set to True or False (not possible), and a probability assigned to each one using vkb:qflag_4 and a 2nd-Order proposition (as explained in section A2 below).

vkb:qflag_6

The allowed values of vkb:qflag_6 are “O, S, A, U” for (Objective, Subjective, Alternative and Universal). These distinguish the context of a proposition as belong to one of these 4 “Worlds”. The Objective and Subjective worlds both refer to the real-world. The difference between these two is the difference between “fact” and “opinion”, which can be distinguished more specifically by considering the perspective from which the proposition is being made. If what is being stated is asserted or declared to be so (or not so) in the real-world is independent of the observer then it is classed as Objective. If however what is important about the statement is that it is the view of an individual, or group which exists in their mind independently of objective facts then it is qualified as being a statement in the Subjective World. In the Objective World if different assertions/declarations are made about the same subject/predicate/object then this constitutes a problem (a contradiction). Only one of these different statements can be so. Whereas in the Subjective World it is entirely normal for different individuals/groups to have different opinions about the same statements, and this does not constitute a contradiction only a difference of opinion. Note that all observations by people involve their personal perception of the world and could in a different sense be said to be subjective. That is not the sense being established here. The question is whether there is a “correct answer” which exists independently of all observers; or whether the statement is about the views/beliefs/intentions of human actors which do not exist independently of them.

An Alternative World may or may not contain entities that exist in the real-world, but either way the Alternative World has a context which is distinct from the world as it really is or was. Examples of Alternative Worlds are plans, and hypothetical situations. There are likely to be many Alternative Worlds so when a 1st-order proposition is stored in a knowledge-base it needs to have an associated 2nd-order proposition which indicates which alternative-world is the context for this proposition.

The Universal World is a space where knowledge is recorded about what is possible to exist in the real world and alternative worlds, and what the possible, impossible, necessary and contingent relations are among what is possible to exist. Such knowledge is about classes of entities. It is ontological.

Default values.

Legacy knowledge-bases, and those established to record simplistic facts, need not explicitly store any or all the qualifiers. When interpreting the contents of such knowledge-bases the following default values may be applied to any missing qualifiers.

vkb:qflag_1 = T (True)
vkb:qflag_2 = A (Asserted)
vkb:qflag_3 = L (Latest)
vkb:qflag_4 = C (Categorical)
vkb:qflag_5 = S (Simple)
vkb:qflag_6 = O (Objective).

What must not be done is to attempt to record a statement in a knowledge-base that does not have the capability to store a given qualifier if the value that should be recorded is not the default value.

A2. 2nd-Order Propositions

A second-order proposition makes a proposition about a set of first-order propositions[13]. Some of the most important uses of 2nd-Order Propositions are:

- To describe the conditions under which the referenced first-order proposition(s) are valid. For example at a given point in time, or for a given time interval; or before after or between given events. (Other forms of validity constraint will be necessary that are not time-related.) Or, conversely, to assert that the proposition is asserted to be true or false unconditionally.
- To link statements qualified as being "Alternative World" with a specific instance of such a world as an object, upon which other facts can be predicated including labels (e.g. Plan-A, Plan-B), provenance etc.
- To provide additional information about the referenced first-order propositions(s), e.g. its source (who asserted it), provenance (e.g. a URL+ sentence number giving access to the full text containing the proposition), perishability, sensitivity (e.g. security classification, releaseability), accuracy, and any confidence-level (or probability) provided with the data as part of the input.
- To affirm that a given person, group/community, or automated assessment process, believes the referenced first-order proposition to be true or false, with or without a level of confidence. This is how different points of view about the same proposition are supported, e.g. A believes xyz, where xyz is a 1st-Order proposition, could be recorded as a Relation='believed_by'.
- To record other "attitudes" (other than belief) towards a proposition by someone, e.g. "A hopes that xyz" where xyz is a 1st-order proposition. This would be recorded as a Relation='hoped_for_by', Entity-class='person', Instance-ID=abc (pointing to A).
- To form a logical combination of first-order propositions, e.g. to express that Proposition A OR Proposition B is true.
- To link an Implied 1st order statement to its operands (the statements that it is conditional on and what the condition is among them).

There will be a need to standardise the predicates used in 2nd-Order Propositions on an Enterprise-Wide basis. The list below is not a list of these predicates: rather it identifies certain classes of predicate within which a number of predicate labels will be needed.

- Validity Condition (e.g. after a certain time, over a time interval, …)
- Source
- Reference
- Accuracy
- Probability
- Sensitivity
- Belief/Confidence
- Delivery Channel

[13] 2nd-Order propositions are literally meta-data (data-about-data) however they are not here called meta-data because the term meta-data is commonly used to refer to data about information-objects (documents, messages, files, images, maps etc.). These "information objects" are the containers which contain data, and meta-data is traditionally predicated on these containers and not on the "contents" of the containers which is what these 2nd-order propositions are doing – where the "contents" are propositions, or sets thereof.

- Specifications of logical combinations of propositions in a set.

A second order proposition consists of an instance of a set, whose members are 1st-order propositions, upon which is predicated an object or a literal. This is illustrated in Figure A-2. There are several different kinds of set (defined below) – one of these must be associated with the above set-instance. A set can have one or more members (each of which is a reified statement, with qualifier-flags).

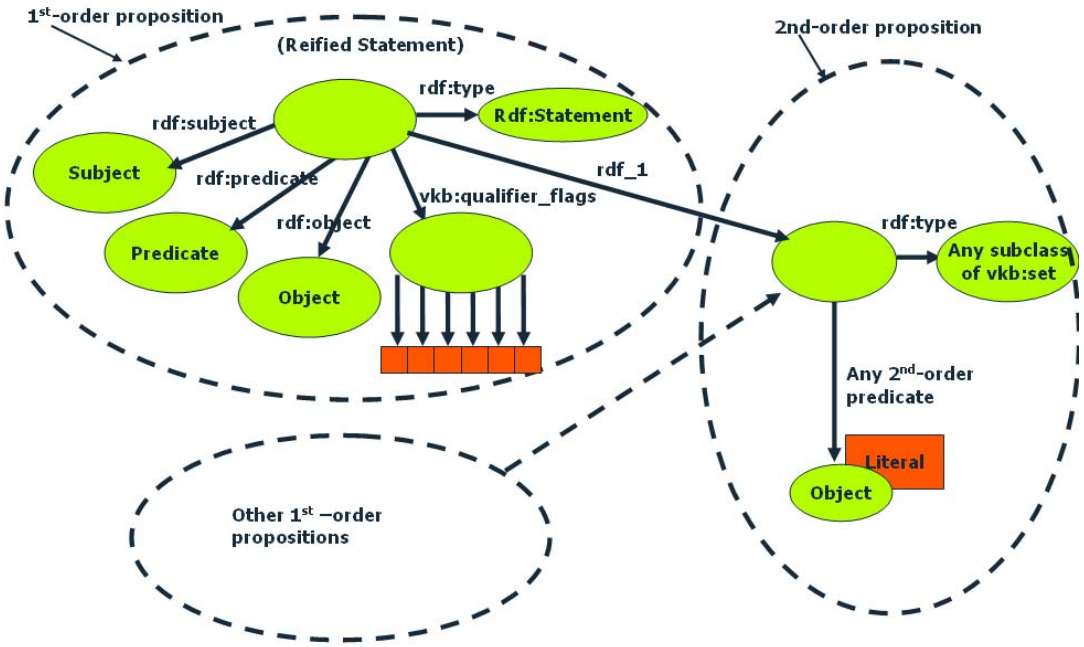

Figure A-2: 2nd-Order proposition

A3. Sets

Sets are used as a means of referring to groups of propositions, normally to define the subject of a 2nd-Order proposition. Sets can also be used to refer to groups of entity-instances, which are then used to define the subject or object of a 1st-order proposition. Each set-instance has a unique identifier (see A4).

Seven different kinds of set have been defined:

1. Primitive sets. Members are propositions.
2. Supersets. Members are Supersets or Primitive Sets
3. Mutually exclusive sets. Members are of types (1), (2) or (3).
4. Equivalence sets. Members are usually Class-instances, but could also be sets of types (1),(2),(3).
5. Logical sets. Members are Supersets or Primitive Sets
6. Ordered sets (Lists). Members are Class-instances.
7. Extension sets: Members are class-instances, with membership conditions.

Primitive Sets. A primitive set is a set of 1st-Order propositions. Members of the set are identified by predicates labelled rdf:_1, rdf_2 etc., but the order of the members is not significant.

Supersets. These are formed by creating sets whose members are primitive sets, or other supersets. The order of the members is not significant.

Mutually exclusive sets. These are sets whose members have the property that only one of them can be so. The members can be primitive sets, supersets or mutually exclusive sets. Mixing supersets and mutually exclusive sets causes a semantic ambiguity as to which elements are mutually exclusive – the sets or their logical members? It is resolved here that the mutually exclusive elements are the primitive propositions that are found by "expanding" all the supersets and mutually exclusive sets to determine their primitive contents, i.e. all the set definitions are logically "flattened" to create set of mutually exclusive primitive sets.

Mutually exclusive sets are especially useful for dealing with ambiguities in the interpretation of input-data. Here each interpretation is recorded as being a different member of a mutually exclusive set. This allows a process of elimination to be applied to determine which interpretation is valid. Mutually exclusive sets and the associated process of elimination may also be used more generally to resolve uncertainties not just in the literal interpretation of data (information extraction) but also in the assessment process – especially in connection with the assessment of evidence (e.g. to determine cause-of-effect or guilt), and the evaluation of alternative courses of action.

Equivalence sets. These are primarily used to record that the fact that different entity-instances refer to the same the same instance of a physical or abstract entity in a World. They can also be used to assert the equivalence between sets of any of the kinds defined above.

The most common situation in which this will be needed in a VKB is to record the fact that instances of entities that have been entered with difference instance-IDs refer to the same entity. This will commonly arise for two reasons:

(a) As a result of a correlation process, where initially the correlation was not recognised and so a new instance-ID allocated.
(b) As a result of an intentional action to limit the sharing of knowledge between different communities/organisations for national security or privacy reasons. Here each community may record knowledge in its own instance of a VKB using different instance-IDs, and knowledge of the correspondence between these instances may be limited among these communities, or only known to yet other communities. The knowledge of these equivalences may need to be protected at, or above, the level of protection accorded to the knowledge in each VKB-instance.

In MOD the most common situation that (b) will arise is to separate information about the attributes of entities that are sensitive due to their ability to identify or otherwise compromise intelligence sources and capabilities, from attributes about the same entities that are less sensitive. These can be stored in different VKBs with different instance-IDs, and the correlation between them only known by the intelligence community. This then avoids the problem of "releasing" information from a highly classified security domain to a lower-classified one – in the following manner … The intelligence community would record its sensitive information about given entities in the highly classified domain, and its less sensitive information (that it wishes to be visible to other communities) about these same entities in a lower-classified domain with different instance-IDs in each domain and record their equivalence in the higher-domain only. There can then be put in place a guard which only allows queries from the higher-classified domain to the lower-classified domain about specified entity-instances-IDs, which blocks such exchanges from low-to-high, and blocks all other types of *information exchange* in both directions. This would enable the higher-classified domain to reason

over all the knowledge about an entity in both domains, whereas in the lower-classified domain access would be limited to the less sensitive information in that domain.

Combined Sets. These sets are provided in order that 2nd-order propositions may be made to state something about their combined result, independently of propositions about the individual set members. For example to assert that Statement A OR Statement B is true, which needs to be recorded independently of assertions/declarations of whether A, B are each true. (This distinction is made through qflag_5, which is set to S= Simple for the individual statements and C=Combined for statements used to specify a combination). Arithmetic or temporal combinations and conditions based on them can also be stated, e.g. that the total of the values in a set of statements is a certain value, or is greater/less than a value; and temporal conditions e.g. that that A is before B.

The operators used to form the combination and their operands (set members) are specified through a 2nd-Order proposition using one of the following predicates.

vkb:logical_combination
vkb:arithmetic_combination
vkb:temporal_combination

The object of these predicates is a literal which specifies in reverse polish notation how the set members should be combined. For example, for a logical_combination:

Object (literal): [1, 2, OR, 3, 4, OR, AND] (for the predicate vkb:logical_combination)
Object (literal): [1, 2, +, 3, 4, +, x] (for the predicate vkb:arithmetic_combination)
Object (literal): [1, 2, Before, 3, 4, After, AND] (for the predicate vkb:temporal_combination)

Where the numbers 1, 2, 3, 4 are the serial numbers of the members of the combined set.

Temporal combinations yield a Boolean logical result and can be mixed with logical operators – the above example evaluates whether: (1 is before 2) AND (3 is after 4) the result being True or False.

To specify a condition related to the combination of set members additional predicates need to be applied to the combined set.

vkb: condition_relation - with its object literal values of { = , < , >, <>, >=, <= }
vkb: condition_value - with its object literal values {T (True), F (False), or a number,}

Ordered Sets. An ordered set is a list of entity-instances. Lists can be made of entities by listing their entity-IDs in order and assigning this list an ID (ordered set). 1nd-order propositions can then be made about the ordered set (i.e. a list), for example that a target-list is approved.

Extension Sets. Extension sets are formed by nominating a set of one or more classes, and then defining a set of zero or more conditions which instances of these classes must satisfy to be considered members of the extension set. The extension set may also be specified as the union or intersection the results of the conditional selection on each class's instances. (This method of defining sets used extensively in OWL.)

A4. Identifying Propositions and Sets

Propositions and sets of propositions need identifiers that are unique within a knowledge-base. This facilitates the predicating of 2nd-order propositions on 1st-order propositions. (In RDF this be readily accomplished by assigning a unique NodeID to the reified statement, or set of statements, which are shown on Figures A-1 and A-2 as “blank-nodes”.)

A5. Identifying Subjects/Objects

Subjects and Objects in statements need to be uniquely identified. Within a given Virtual Knowledge Base there needs to be a consistent naming convention and method for ensuring uniqueness. It is here proposed that the convention be as follows: namespace: class #instance-ID, for example:

myvkb:target#123456

This indicates that this object is an instance of class “target”, in namespace “myvkb”, and is assigned the ID “123456”. Note that the instance-ID part does not have any “meaning”. Instance-IDs should not be names or have any significance other than distinguishing one instance of a class from another.

If it is required to refer to the whole class and not an instance then the “#” and instance-ID are omitted.

(Some Objects are literals and these are not applicable to the above convention.)