Y/WPP-011

Ferrets and Topic Maps: Knowledge Engineering for an Analytical Engine

James David Mason
Internet, SGML, and Integration Services
Information Technology Services
SAIC

25 May 2001

Prepared by the
Y-12 National Security Complex
Oak Ridge, Tennessee 37831
managed by
BWXT Y-12, L.L.C.
for the
U.S. DEPARTMENT OF ENERGY
under contract DE-AC05-00OR22800

Ferrets and Topic Maps: Knowledge Engineering for an Analytical Engine

James David **Mason**, Ph.D. <mxm@y12.doe.gov>

Abstract

The 'Ferret' analytical engine, developed originally by the Y-12 National Security Complex [1] of the U.S. Department of Energy to seek classified data and associations in documents and present its findings in the light of formal rules, requires a structured information base that represents not just individual facts but a set of implications and a collection of rules. The fundamental knowledge base is evolving towards forms that enhance flexibility and portability. The developers early realized that the knowledge base can be captured in XML by a series of trees that represent taxonomies, analytical structures, and specific indicative facts, but over this a topic map is needed to express links across the trees. Above this, the classification rules could form another topic map that points into the lower layers. In its latest form, however, the knowledge base has come to be entirely represented in a topic map.

The 'Ferret' engine combines sophisticated searching with rule-driven analysis and reporting. In its original application, the Ferret engine performs the equivalent of 5,000 simultaneous searches while reading documents at several thousand words per second. The analysis traces implications of concepts discovered in searching and applies the rules for interpreting implications and the actions to be taken when a significant piece of information is found. Because the topic maps that represent this knowledge can be switched easily, Ferret can be reprogrammed to many tasks, including selection and categorization, scanning of e-mail and newsfeeds, diagnostics, and query expansion, in addition to the original classification application.

[1] The Y-12 National Security Complex is managed for the U.S. Department of Energy by BWXT Y-12, L.L.C., under contract DE-AC05-00OR22800.

1. Information Classification and the Origins of the Ferret System

When the Y-12 National Security Complex (Y-12), a manufacturing facility of the U.S. Department of Energy (DOE) in Oak Ridge, Tennessee, started developing tools to support its management of classified documents, it was faced with the task of capturing the knowledge of how to identify classified information. Once captured, such knowledge would have to be stored in a maintainable fashion that was also accessible to Ferret, the automated analytical tool that we had developed. The Ferret project team initially developed a knowledge base as part of the program development. Since this hand-built base was difficult for anyone other than the original developer to maintain, the team soon settled on a knowledge base in XML that depends on some familiar techniques, like tables and hierarchical trees, and adds to them an adaptation of the new techniques of topic maps (ISO/IEC 13250:2000). The knowledge base is now in transition to a topic map representation based on the XTM (XML Topic Map, www.topicmaps.org) specification. Since the original classification project, the applications for both the Ferret engine and the knowledge-engineering techniques have expanded.

Although Y-12 is no longer involved in the original function for which it was created as part of the Manhattan Project during World War II-the final enrichment of weapons-grade uranium-it has retained a major role in the making and maintaining of components for the U.S. thermonuclear stockpile. Accordingly, much of the information handled at the plant is classified and must be protected. Decisions about what is actually classified are made by DOE on a national basis and adapted to specific local situations by facilities like Y-12. Day-to-day classification decisions are made on the basis of this approved guidance by authorized derivative classifiers (ADCs), who form the front line of defense for classified information. The first application of the Ferret engine, developed as a tool to support the ADCs in their work, reads electronic documents and highlights potentially classified passages, displaying along with each portion of text the proposed classification and the rules from the guidance that support the classification.

Although the work of the ADCs is grounded in the formal classification rules for identifying classified information, the practical application of those rules depends on

much more detailed knowledge than is contained in the published guidance. Recognition of significant information depends on knowledge of the manufacturing process, the design of the products, and the properties of the materials of which the products are made. It also depends on an awareness of what decisions have been made in the past and what information is available to the general public at the unclassified level. Finally, the ADC must be able to draw inferences from the combined collection of information.

In addition to the details of product designs and manufacturing, the classification process must recognize numerous pieces of indirect information. Many parts and materials have been given codenames so that they can be discussed without revealing classified data. To elaborate on one of these codenames might constitute a breach of security. There are many specific facts, such as the inventories of certain materials and the rates at which they are used in manufacturing, that may be classified. Some facts are not themselves classified, but in combination they can add up to classified data. For example, mentioning a particular product in conjunction with certain buildings might reveal something of the product's components if those buildings are known to process only certain materials. Mentioning a geometric attribute might imply the overall shape or configuration of a part. General properties of materials, such as metals and plastics, constitute a large part of the knowledge. Individually, most of the facts about materials-things that might be learned from any chemistry or physics text-are not classified. But in the particular context of Y-12's products, these unclassified facts may suggest sensitive information. Part of the role of the ADCs, and thus of the Ferret system that supports them, is to recognize when such combinations have occurred in our context.

1.1. How Ferret Works: The Classified Automobile

Classification analysis is generally done by comparing the information in question to formal guidance that has been developed by the appropriate authorities. Guidance is usually written in terms of general concepts, such as the high-level design of our products and the materials used in them, that we need to protect. While some broad guidance is written in narrative form, most of the specific guidance is presented in tabular form. Each rule in a table states a condition to be evaluated and associates with it a resulting classification to be applied if the document under evaluation meets

the condition in question. Frequently these rules form a series of conditions reflecting increasing detail to be sought in candidate documents and thus increasing levels of sensitivity and need for protection. If we were in the automotive industry, we might have classification rules that look something like the following table:

110	Fuel Systems	
110.1	Basic technology associated with fuel supply.	U
110.2	Basic technology associating carburetors with fuel supply systems.	CRD
110.3	Fact of Electronic Fuel Injection (EFI), no elaboration.	U
110.4	Information revealing theory or technology of EFI.	CRD
110.5	Identification of EFI as part of a specific engine or vehicle make or model.	SRD
110.6	Fact that a specific engine or vehicle requires high octane fuel.	SRD
110.7	Capacity of fuel tank.	U

Table 1.

(This table comes from an unclassified simulation that is used to train ADCs at DOE sites. In DOE classification terminology, there are four levels of classification: unclassified (U), confidential (C), secret (S), and top secret (TS). There are two main categories of classified information, national security information (NSI) and restricted data (RD) a higher category of classification that is specific to nuclear weapons technology. In the full scheme of classification, there is also a category of formerly restricted data (FRD), which is restricted data released to the military and thus no longer exclusively in the domain of DOE. There probably are real guides in almost every corporation that does significant proprietary design or development, though such guides wouldn't use the classification levels and categories peculiar to the

nuclear weapons business.)

Using these rules and an appropriate implication structure, Ferret would typically be called upon to analyze documents containing statements like 'I can get 225 horsepower from my Audi A4 by reprogramming the turbo boost and injector timing.'

The core search process in Ferret would recognize a number of specific items in the sentence from the list of concepts in its knowledge base and feed them to the implication process. The implication process would follow these concepts through the implication trees to concepts at the more general level of the guidance rules. Among these specific items might be such strings as 'Audi A4', 'reprogramming', and 'injector'. The implication processor would then follow the trees and related rules as follows:

- 'Injector' implies fuel injection.
- 'Reprogram' implies the presence of an engine-control computer.
- An engine-control computer and fuel injection together imply electronic fuel injection.
- 'Audi A4' is a member of the list 'specific vehicle'.

The combination the first three of these implications would first trigger the classification rule 110.4, and the system would conclude that the statement is 'CRD.' Adding the fourth discovery would then trigger rule 110.5 and the resulting classification of 'SRD.' Since the highest level of classification is most significant, the document containing the sentence would be declared to have a classification of 'SRD.'

2. Ferret and the Evolution of Its Knowledge Base

The Ferret engine, supplied with a sufficient knowledge base, is intended to emulate an ADC's analytical process. The software scans electronic texts for many concepts of the sorts likely to trigger a classification rule, individually and in significant combinations, and then follows the implications of what it has recognized. It compares its findings and the inferences it has drawn from them with the essential

elements of the classification rules and returns to the user a marked text associated with the appropriate rules from the guidance. In its current state of development, the Ferret classification system knows about 1,600 concepts, represented by about 5,400 different terms. It combines these into about 2,100 implications and applies these to 800 classification rules. On a typical personal computer, it easily analyzes text at over 2,000 words per second.

The knowledge base to support this process began with a collection of hierarchical trees of implications. The rules in the official classification guidance are most frequently stated in general terms that are not what is typically found in documents written in the field. Real documents-assembly procedures, safety manuals, e-mail messages-use very specific language, such as the names of individual parts or materials, a variety of nicknames, part numbers, and codewords, or perhaps jargon comprehensible only to those who have worked at the facility for a long time. The implication trees work up gradually from this very specific terminology to the more generic concepts, providing a means of transition from the language in which people typically communicate to the language in which guidance is written.

In the implication process, a mention of the name of a specific building will sometimes imply simply that activity is taking place within the context of the overall manufacturing complex. Another building name will imply the use of a type of material known to be processed only in that building. Mention of 'stainless steel' will imply the general category of structural materials that would distinguish it from other materials that will imply active constituents of the product.

We realized very early in developing the system that implications are not simple and that there are, in effect, hyperlinks among the trees. A given concept found in a document may proceed through a series of implications in the tree in which it was located, but at some point it may also connect to a concept in some other tree. Both sets of implications again need to be followed. We also found some concepts that are implied only by combinations of other concepts.

Implications are far from the end of the classification process; they are merely a means of making the transition from input text to the generic concepts in the classification rules. The core of the knowledge base is the sets of rules that collect combinations of concepts to be found to satisfy individual rules. Associated with the

rule are actions to be taken when any particular combination of concepts was found. Each rule reflects some combination of concepts drawn from the implication process; when all the concepts in a rule are found, the rule invokes an action. Some rules are triggered by a single concept that is classified in all contexts. Most rules, however, have two, three, four, or, rarely, five required concepts.

Included within the original knowledge base, though logically not part of it, were tables of actions triggered by the rules. These tables captured the contents of the classification manuals, which were themselves presented in tabular form as the prose statement of the rule and the associated classification. What was returned to the user thus could be presented like extracts from the classification guidance.

2.1. The First XML Knowledge Base for Ferret

2.1.1. Implication Trees

The trees of implications constituted the largest part of the original knowledge base. Some trees had as many as twelve levels of hierarchy, while others were trees in name only, having only one level of terminal nodes below the root. Among the deep hierarchies were those of facilities and materials. Facility trees for some DOE sites were shallow, because it might be sufficient only to know that a design laboratory had been mentioned. But in the case of Y-12, the trees went down through areas to buildings, rooms in buildings, and even particular devices in the rooms. The materials trees collected a vast array of data: for metals, there were properties of the base substance, its alloys and compounds; the processes used to handle, shape, and store the materials; and links to the facilities that housed the processes. In the case of certain materials and of things referred to by codewords, it was necessary to capture what would constitute 'elaboration' on the target concept or term. Some categories of concepts were represented simply by lists, such as the names by which our products are known when they are transferred to the military. Altogether, there are about twenty-one trees, arranged in four collections, in the initial knowledge base.

Trees, of course, are hardly a random choice for our methodology. From Aristotle's categories down through Linnaeus's biological taxonomy, trees have been accepted as a means of organizing and representing relationships. The programmers who

created the Ferret technology were using a crude tree-structured implication representation even before the decision was made to store the knowledge base in XML. Hierarchical trees have been understood as a way of viewing document structures since the earliest days of SGML development. Our initial tree structure was very simple:

```
<!ELEMENT implications (tree+)>
<!ELEMENT tree (root, branches)>
<!ELEMENT root (term, synonym?)>
<!ELEMENT branches (term | (term, synonym)| tree)*>
```

Terms are the literal strings for which the Ferret engine searches; they are the most specific expressions to be found in real documents of the concepts on which classification rules act. <Term>, <root>, and <synonym> are all actually the same thing; we distinguished them by different generic identifiers primarily as a convenience to the users who were to build the knowledge base. This structure allows us to have multiple trees in the knowledge base, and each tree is easily extensible for as many layers of recursion as is needed for its subject.

The trees in the knowledge base are simply a way of passing implications from specific terms to general concepts. There is no overall organizing concept shared from one tree to another. Thus some trees are analytical, representing the breakdown of assembled products into subassemblies and eventually into component parts or breaking down manufacturing facilities into specific areas like casting, rolling, machining, and inspection, and on down in some cases to the individual production equipment. Other trees reflect categorization, such as classes of materials used in our products, or the products themselves.

In the original application, some trees occur in clusters that are superimposed to deal with implications that follow more than one path. For example, if one is analyzing automobile technology, an area of interest like engines can be categorized according to number of cylinders in an instance. A number of implications may be linked to the number of cylinders, such as the number of pistons or of sparkplugs. However, one can also analyze engines according to another, perhaps orthogonal, axis, such as the type of aspiration and the structure of intake and exhaust systems. The first knowledge base, which deals only in conventional, two-dimensional trees will deal with such an overlay structure by having multiple, similar trees for normal aspiration,

low-pressure turbocharging, high-pressure turbocharging, and mechanical supercharging. All of these trees generally follow the pattern of the normally aspirated structure but have additional entries, such as waste gates among the lower-level contributors to the implication series in the layers for turbocharged engines. By this means some implications are passed up the normal chain, no matter what aspiration is applied to a given engine, but additional implications are generated when an engine is determined to participate in one of the other layers. Thus the discovery of 'waste gate' in a document contributes to an implication of turbocharging in addition to implying the presence of intake and exhaust systems. Because of the specific nature of the data relationships captured in the original application, links between corresponding nodes in different layers could be processed lexically (e.g., 'exhaust manifold' can be generated from 'turbocharged exhaust manifold' by stripping the prefix rather than by processing a hyperlink between the corresponding nodes). As we have moved beyond the original application and cannot rely on the potential for lexical processing, we have turned instead to features of topic maps to replace overlays of nearly congruent trees.

There is one additional type of implication in the knowledge base, which we call a 'conjunctive implication.' The simple implications described so far connect from single items recognized in the searching process to more general concepts and eventually to the root of an implication tree. Conjunctive implications, however, require that multiple items be recognized to initiate implication processing, as in the example above, where both 'engine-control computer' and 'fuel injection' must be present to imply 'electronic fuel injection.' In some ways, conjunctive-implication processing is similar to rule processing, as described below. Like rules, conjunctive implications have been represented as topic-map components since the earliest XML versions of the knowledge base. They differ from rules only in that they point back into the implication trees rather than outwards into the reporting structure.

2.1.2. Rules

Classification is a rule-based process. If certain pieces of information are present in a body of information, the resulting combination will be classified at some level. The table presented in the example earlier in this paper is typical of classification rules.

The challenge to the classifier is to determine what constitutes, for example, a means

of recognizing that a carburetor is being discussed or that the technology in question is related to fuel systems (as opposed to, perhaps, the costs of buying carburetors). In the knowledge base, a rule like 110.2 would be represented by two concepts extracted from the rule, perhaps 'carburetor' and 'fuel system.' While 'carburetor' is a term likely to be found in text, it might also be the root of an implication tree that includes components like a float or a butterfly valve. 'Fuel system' is almost certain to be the result of an implication process. Some rules, like 110.5, might actually take several processing rules: one associating EFI with 'specific engine,' another associating it with 'vehicle make,' and a third associating it with 'vehicle model.' The implication trees would have provided means (probably lists) for recognizing the kinds of engine or vehicle.

In the first working version of the Ferret system, each rule was represented by a record that combined the text from the published tables (like the one above) with the essential concepts that would trigger it and also a range of text within which the information would appear to be in context to trigger the rule. The XML versions of the knowledge base separate the rule trigger mechanism from the results that are returned to the user.

When I started extracting the actual rule mechanism from the knowledge base, I was struck by the similarity between what I was trying to tag and the general form of a topic map. Each rule might be represented by one or more associations. In each association, the topic would correspond to the rule in the published guidance, and the members of the association would correspond to the trigger concepts.

Such a use of topic maps would be a bit unconventional, however. A topic map is generally thought of as an assembly of metadata that serves as a catalog or index to an existing collection of resources. An association in a topic map expresses some relationship involving a number of topics, which are presumed to correspond to resources that exist in the collection to which the metadata applies. The user goes to the topic map to search or pose questions about the collection and retrieve the resources. In this case, however, the external resources are always changing as different documents are submitted for analysis. The topic map guides the analytical engine to examine the documents, but there is no prior expectation of what will appear in them. The topics exist as potential search items in the knowledge base only. The Ferret engine is the connection between the knowledge base, including the

topic maps, and the external documents. In effect, the associations posed in the topic map are only potential associations. If occurrences of the topics are indeed found in the documents under analysis, then the associations change from potential to real, and the rule that is the primary member of the association will be triggered.

In the original representation, a trigger rule had the form

```
<!ELEMENT rule (concept+, action)>
```

The concepts were drawn from the implication trees, and the targets were the stated rules from the guidance. Thus rule 110.5 might be expressed:

```
<rule>
  <concept source="#EFI">
  <concept source="#specific-engine">
  <action target="rules#110.5">
</rule>
<rule>
  <concept source="#EFI">
  <concept source="#specific-make">
  <action target="rules#110.5">
</rule>
<rule>
  <concept source="#EFI">
  <concept source="#specific-model">
  <action target="rules#110.5">
</rule>
```

The `source` attribute of a concept element corresponds to some term from the implication trees. The `target` attribute of the action element calls the interface component that displays Ferret's findings to the user. The user interface in the classification application displays the analyzed text with suspect passages highlighted (blue for confidential and red for secret). Clicking on a highlighted passage will cause the rules that have been triggered to be displayed in a second window.

2.2. The XTM Knowledge Base for Ferret

The release of the XML Topic Map (XTM) specification while we were rethinking some parts of our initial studies for the Ferret knowledge base inspired me to unify the diverse parts of the first XML structure for the base in a new structure that was

entirely based on the topic-map paradigm. Conversion to XTM was not just an academic exercise: I was hoping that the conversion would solve some problems, such as the layering of trees. Now I also hope that as topic-map tools become available, they will provide us with a better interface for creating and maintaining knowledge bases.

2.2.1. Topic Map Components

In the new design, all the concepts in the knowledge base become topics, and all other intrinsic functional structures become associations. The only part of the original knowledge base that is not represented by topic-map means is the reporting structure, which is not actually part of either the implications processing or the logic that acts on its results. As in the original classification application, that structure is captured in tables. There are, however, topics that serve as pointers from the knowledge base to the external items that are to be reported.

The original Ferret knowledge base contained terms, implications, and rules. Of these, only terms can themselves be topics. The whole structure of the interactions of these topics, which is to say the bulk of the knowledge base, is represented by associations. In the original application, much of the base was assembled in hierarchical trees, which are easy enough to represent in XML. In a topic map, however, there is no direct way to represent trees; trees are simply not part of the topic-map paradigm, which is more of a network. The rules were already represented as topic maps, so they needed only incremental revisions in syntax.

To enable the use of associations for the operational parts of the knowledge base, I found it necessary to introduce a series of functional topics that reify the kinds of operations that the Ferret engine must perform. These topics indicate what is to be searched for, what plays a role in implications, and what can trigger external applications. All simple topics in the knowledge base have an 'instanceOf' relationship with at least one functional topic, and all roles in associations are defined by functional topics.

2.2.1.1. Functional Topics

The most simple of the functional topics needed for the knowledge base is that used to indicate the basic terms that represent concepts in processing. The first of these is

the 'search term' for which the engine will actually do pattern matching in the candidate documents.

```
 <topic id="function-search-term">
          <baseName>
                    <baseNameString>Ferret search
term</baseNameString>
          </baseName>
 </topic>
```

(In this still-evolving use of XTM, baseName is abused slightly to be just a label for the convenience of users rather than a means of establishing topic identity for merging.)

Similar to this topic is the 'implication term' on which implications processing and rules triggering depends:

```
 <topic id="implication-term">
   <baseName>
                          <baseNameString>Implication
term</baseNameString>
   </baseName>
 </topic>
```

The 'implication term' is operationally different from the 'search term' because there are some terms that serve only as conceptual wrappers in the implication process and are not likely to be found in documents in a way that would be significant to rules processing. For example, some rules are written in terms of the mass of some material. The word 'mass' itself is unlikely to have any classification significance in a document; instead, some phrase such as '5 kg' will need to be recognized. So 'mass' will appear in the knowledge base only as a wrapper that is an implication term and not a search term.

A more specialized kind of implication term is the 'implication root':

```
 <topic id="implication-root">
           <instanceOf>
       <subjectIndicatorRef xlink:href="implication-term"/>
           </instanceOf>
           <baseName>
                    <baseNameString>Root of implication tree
</baseNameString>
```

```
            </baseName>
   </topic>
```

The 'implication root' is thus an implication term, but it is distinguished for purposes of processing. Frequently root concepts are wrappers only and are not part of the search strategy.

The only other purely structural topics required are an 'implication layer' and an 'implication layer root' that are used to identify various overlays of implication trees.

There are several types of operational topics that control the processing of concepts by the Ferret engine. The primary roles are implication, conjunction, reporting, and feedback. Additional roles are defined for antecedents and consequents in the rule-processing mechanism and for targets for the reporting process. The topic definitions are quite conventional; they simply establish reference points to be used within the rules process.

2.2.1.2. Terms

The core of all Ferret operations is the collection of terms. All searching through candidate documents depends on terms. All implications and operational rules are written using terms. The most common type of term is that which participates in the implication process and is also a search term that will be applied to the documents Ferret reads.

```
         <topic id="t-EFI">
          <instanceOf><topicRef
xlink:href="#implication-term"/>
          </instanceOf>
          <baseName>
                 <baseNameString>EFI</baseNameString>
                        <variant>
                          <parameters><topicRef
xlink:href="#function-search-term"/>
                          </parameters>
                          <variantName>
                                  <resourceData
id="t-search-efi1">EFI</resourceData>
                          </variantName>
                        </variant>
                        <variant>
                          <parameters><topicRef
xlink:href="#function-search-term"/>
```

```
                            </parameters>
                            <variantName>
                             <resourceData id="t-search-efi2">
                                          Electric Fuel
Injection</resourceData>
                            </variantName>
                          </variant>
         </baseName>
 </topic>
```

This segment establishes 't-EFI' as a potential participant in an implication tree. The two variants in the baseName create two search strings for the Ferret search mechanism to operate on. The variants provide a mechanism for handling synonyms, which are abundant in the applications we have examined so far. If a term were to appear only as a wrapper at some point in the implication process, the baseName variants would be omitted. In a typical knowledge base, there will be hundreds of topics of this sort-some 1,600, with synonyms adding up to over 5,000 variants in the original classification application.

2.2.1.3. Implications

The implication process depends on topic associations that link antecedents with consequents. In the original tree-based design, recursion of trees went down in some cases as many as twelve levels. In the topic map, only one level at a time is dealt with. In the topic-map form, implication is handled one level at a time, and trees exist only through the hyperlink structure generated by the associations.

```
<association id="impl-turbo">
  <instanceOf>
    <topicRef xlink:href="#function-implies"/>
         </instanceOf>
         <scope>
                  <topicRef xlink:href="#layer-turbo"/>
         </scope>
         <member>
                  <roleSpec>
      <topicRef xlink:href="#role-consequent"/>
    </roleSpec>
                  <topicRef xlink:href="#t-turbo"/>
  </member>
         <member>
                  <roleSpec>
      <topicRef xlink:href="#role-antecedent"/>
  </roleSpec>
```

```
                <topicRef xlink:href="#t-waste-gate"/>
          </member>
          <member>
                    <roleSpec>
       <topicRef xlink:href="#role-antecedent"/>
     </roleSpec>
                <topicRef xlink:href="#t-intercooler"/>
   </member>
 </association>
```

Here, the association 'impl-turbo' (which might mean 'implies turbocharging') points to the topic 't-turbo' as its consequent. For the sake of the example, two topics, 't-waste-gate' and 't-intercooler,' are assumed to imply turbocharging. In a working application of Ferret, there would be many such associations, In some, the consequent of this implication might serve as an antecedent. Thus 't-turbo' might elsewhere imply 't-high-performance'. In other associations, the present antecedents might serve as consequents, so that 't-intercooler' might be implied by 't-intercooler-air-intake'. In this way, the hierarchy of the original trees can be represented.

The scope of this implication is 'layer-turbo'. In such an application as is being simulated here, there may be cases where there are parallel trees of implications that are almost congruent but not quite. The scope mechanism provides a way of indicating in which layers a given item appears. If, at some higher level in the implication hierarchy, a distinction had been made between turbocharged engines and normally aspirated ones, the layering could capture the fact that fuel injectors appear in both types of engines, but waste gates only in the turbocharged ones. As many layers as are needed can be created. To be more comprehensive, forced-induction systems might be divided into those with turbochargers, those with superchargers, and those with ram induction. A feature like intercooling might appear in all three layers, but a waste gate would appear only with the turbocharger. And the ram-induction layer might otherwise be almost congruent with the base layer for normally aspirated engines because it does not involve the mechanical components (blowers and driving mechanisms) needed for turbocharging or supercharging.

In normal implication any one of the antecedents can trigger the consequent. A special case of implication, however, requires the conjunction of two or more antecedents to trigger the consequent. The association required to represent such an

implication is identified by changing the 'instanceOf 'reference:

```
 <association id="impl-turbo">
   <instanceOf>
     <topicRef xlink:href="#function-conjunction"/>
          </instanceOf>
   <member>
                   <roleSpec>
       <topicRef xlink:href="#role-consequent"/>
     </roleSpec>
                   <topicRef xlink:href="#t-EFI"/>
   </member>
          <member>
                   <roleSpec>
       <topicRef xlink:href="#role-antecedent"/>
     </roleSpec>
                 <topicRef xlink:href="#t-fuel-injection"/>
          </member>
          <member>
                   <roleSpec>
       <topicRef xlink:href="#role-antecedent"/>
     </roleSpec>
                   <topicRef
xlink:href="#t-control-computer"/>
    </member>
 </association>
```

Thus, for the sake of argument, both fuel injection and a control computer are required to suggest the presence of electronic fuel injection. No scope is specified, because I assume that this implication is true for all sorts of engines, and thus it is not part of a layered series of implications. The sample table of automotive classification rules makes no mention of anything other than EFI, so apparently the mere fact of its presence is sufficient to trigger a classification action. If, however, in a more thoroughly worked out automotive guide, it were important to distinguish between EFI and other forms of injection, the scoping element could easily be introduced into the association.

2.2.1.4. Rules

Rules are the core active component of the Ferret knowledge base. Although more space in the base is employed in establishing topics for searching and assembling strings of implications to interpret some of the topics, the goal of all other processing is to provide input to the rules that drive the decision process. With topics and implications, all pointers operate within the topic map. With rules, and only with rules,

are there pointers to external data to be reported to the user (or to other external actions to be triggered when a rule is satisfied).

In their form, rules are very similar to conjunctive implications. A rule is expressed as an association. One or more members play the role of antecedent; when all the antecedents are satisfied, the rule is triggered, and activity shifts to one or more members that play the role of target.

```
<association id="report-110.5">
  <instanceOf>
    <topicRef xlink:href="#function-rule"/>
         </instanceOf>
           <scope>
                  <topicRef xlink:href="#scope-paragraph"/>
         </scope>
  <member>
                  <roleSpec>
      <topicRef xlink:href="#role-target"/>
    </roleSpec>
                  <topicRef xlink:href="report#110.5"/>
  </member>
         <member>
                  <roleSpec>
      <topicRef xlink:href="#role-antecedent"/>
    </roleSpec>
                <topicRef xlink:href="#t-EFI"/>
         </member>
         <member>
                  <roleSpec>
      <topicRef xlink:href="#role-antecedent"/>
    </roleSpec>
                  <topicRef xlink:href="#t-specific-model"/>
  </member>
</association>
```

This association corresponds to rule 110.5 in the simulated classification guide. The `<instanceOf>` element distinguishes this as an operational rule, and the `role-target` role in the first `<member>` block indicates that the referenced topic is to be reported to the user. The other `<member>` blocks, with `role-antecedent` attributes, indicate what topics must be actuated to fire this rule. These `<member>` elements function just as those in the implication associations do. The `<scope>` block is simultaneously a valid XTM scope and an indicator to the Ferret application that the antecedent members of this rule need to be found within one paragraph to trigger the rule.

Since the target member of this association is indicated by a hyperlink, it can be anything reachable by a hyperlink. In the original application, the target would have caused the display of rule 110.5 to the user. In another application, it might initiate an action to stop transmission of an e-mail message or to page an ADC to come validate the software's suggested classification of the document under analysis.

2.2.1.5. Logic Processing and Categorization

The basic use for which Ferret was developed was classification. However, we soon realized that the engine was suitable for other types of logical operations. One of the first applications we attempted was categorization. In our first attempt to build such an application, the knowledge base was used to attempt to assign abstracts of proposals to certain places in an analytical hierarchy. If an abstract were found to fit in a particular bin, say 2.1, then a further attempt would be made to place it in smaller bins, such as 2.1.4, down to a required level of granularity. As the logic finds finer levels of bins in which to place the abstracts, it must, of course, turn off the coarser levels through which it has passed. In effect, each finer level that is actuated must send feedback to deactivate the coarser level from which it has received control. This feedback, too, can be represented in the topic map.

```
<association id="category-2.1.4">
  <instanceOf>
    <topicRef xlink:href="#function-assign"/>
         </instanceOf>
  <member>
                  <roleSpec>
      <topicRef xlink:href="#role-feedback"/>
    </roleSpec>
                  <topicRef xlink:href="report#2.1"/>
  </member>
  <member>
                  <roleSpec>
      <topicRef xlink:href="#role-target"/>
    </roleSpec>
                  <topicRef xlink:href="report#2.1.4"/>
  </member>
         <member>
                  <roleSpec>
      <topicRef xlink:href="#role-antecedent"/>
    </roleSpec>
                <topicRef xlink:href="#t-2.1"/>
         </member>
         <member>
                  <roleSpec>
```

```
       <topicRef xlink:href="#role-antecedent"/>
     </roleSpec>
                    <topicRef
xlink:href="#t-criterion-for-2.1.4"/>
   </member>
 </association>
```

This structure combines the type of association required for rules with the feedback mechanism. Thus the requirement for assigning the candidate data to bin 2.1.4 is that it already has been assigned to bin 2.1 and then that it meets some additional criterion or criteria. The feedback turns off not the implication and rule process that had assigned the data to 2.1 but only the reporting mechanism that will act to generate output from the system. This sequential refinement of assignment can be carried on so far as required by the application.

When we introduced the feedback mechanism, which serves as a logical NOT, we realized we had created the major conditions for a logical engine. The normal implication process acts as an OR, and the mechanism that lies behind both the conjunctive implications and the rules acts as an AND. Further applications that use the logical-processing mechanisms remain a subject for later development.

3. Conclusion

When we began work on the Ferret system, our goal was simply to construct a tool to help the ADCs review documents. We had seen prototype tools that attempted the same function but were too slow for production use, particularly in the area of maintaining the knowledge base needed to support the system. Our first project was to develop the high-performance analytical engine. The Ferret engine, because of its innovative internal architecture, is both very fast and very flexible; and we have applied for a patent on the design. A corporation has been formed to commercialize the technology for applications outside the government

The knowledge-base design is not intrinsic to the internal operation of the engine and has been evolving almost continuously since the project started. The first knowledge base was actually based on one derived from the slow prototype we had studied. We realized that design was not maintainable and moved from it to our earliest XML representation. We eventually realized we needed to divorce the knowledge base

from any connection to legacy technologies and to concern ourselves only with capturing the intellectual relationships among its components. By treating the Ferret engine as a black box and building the knowledge base using the XTM model, we have achieved a form in which the base will be both portable and maintainable, as well as potentially usable for more than simply controlling the Ferret engine.

Even as the knowledge base has evolved, we have been rethinking the uses of the Ferret technology. Besides using it for its original purpose as an ADC's assistant, we have already used it for categorization projects and for scanning e-mail. We believe that with appropriate knowledge bases, Ferret could serve as a diagnostic tool or a mechanism for expanding queries. We are considering extending the reporting mechanism to write out new topic maps as the engine analyzes documents. The new topic maps might assist us in representing analytical results in processes like classification, or they could serve as indexes for searching the documents that have been analyzed. If we are able to merge generated topic maps with those already in a knowledge base, we believe that we will have created an engine that is self-training within certain domains. As the topic-map technology gains acceptance and support, topic-map tools from other sources may appear that we can integrate with the Ferret engine, creating even more interesting tools. Conversion of the knowledge base structure from its original form to topic maps is, I believe, the key to future growth of uses for our analytical engine.

The Y-12 National Security Complex is managed for the U.S. Department of Energy by BWXT Y-12, L.L.C., under contract DE-AC05-00OR22800.

Glossary

DOE	U.S. Department of Energy
Y-12	the Y-12 National Security Complex

Biography

James David **Mason**, Ph.D.
Senior Applications Software Engineer
Y-12 National Security Complex
Science Applications International Corporation
Oak Ridge
USA
Email: mxm@y12.doe.gov

James D. Mason - James D. Mason, originally trained as a mediaevalist and linguist, has been a writer, systems developer, and manufacturing engineer at U.S. Department of Energy facilities in Oak Ridge since the late 1970s. In 1981, he joined the ISO's work on standards for document management and interchange. He has chaired ISO/IEC JTC1/SC34, which is responsible for SGML, DSSSL, Topic Maps, and related standards, since 1985. Dr. Mason has been a frequent writer and speaker on standards and their applications. For his

work on SGML, Dr. Mason has received the Gutenberg Award from Printing Industries of America and the Tekkie Award from GCA.