Engineering a Model of the Subjective Judgement of Experts
in a Data Fusion Application

Jane O’Brien and Mark Bedworth
Signal Processing and Imagery Department, DERA Malvern, UK
Email: job@signal.dera.gov.uk

Abstract – Airborne anti-submarine warfare includes fusion carried out by the Tactical Navigator when making decisions about the deployment of sonobuoys. This can be automated using a Bayesian Belief Network. However, the effectiveness of such a system depends on the accuracy of modelling the Tactical Navigator's decision-making process. This in turn relies on a practiced Tactical Navigator supplying the correct decisions to make and judgmental information on how these decisions would be affected by other factors. Such knowledge is only gained through experience and is difficult to quantify. Knowledge engineering methodologies and tools are available to aid with such knowledge acquisition and quantification.

This paper will describe the airborne anti-submarine warfare problem and highlight the need for knowledge engineering techniques to develop a successful solution. It will provide a general background to knowledge engineering and describe methodologies, including CommonKADS, for carrying it out. The paper will then detail the application of CommonKADS to the development and implementation of an automated decision-making aid for the Tactical Navigator.

Keywords: Anti-submarine warfare, Bayesian Belief Network, Expert, Information Fusion, Knowledge Engineering

1. Introduction

Airborne Anti-Submarine Warfare (ASW) is a complex military task where many decision-making problems cannot be explicitly solved either theoretically or by taking measurements [1]. One way to study the decision-making process in this case is by developing a computer simulation of the situation. An area where this has been evaluated is in assessing how the Tactical Navigator (TacNav) determines what action to take next when flying a mission. This has been tackled by developing an automated decision aid to the TacNav. The different information available to the TacNav indicates that this is a data fusion problem. In addition to providing an insight into operational problems, this aid can also be used to evaluate the possibility of fully automating the airborne anti-submarine warfare task.

The development of such a system depends on identifying the correct information to fuse. Since part of this information is encapsulated in the TacNav's experience the process is not as simple as it might be and depends on using knowledge engineering (KE) techniques. There are many KE techniques to use; this application has provided an opportunity for the tool, CommonKADS, to be evaluated.

Following this introduction, this paper will describe the ASW application, indicating where the fusion process takes place and highlighting the key modelling issue. It will then describe KE and provide an overview of CommonKADS. A description of the ASW case study using CommonKADS and its implementation will be provided, followed by conclusions on the use of KE techniques in general and CommonKADS in particular.

2. An Anti-Submarine Warfare Task

The aim of the ASW mission under examination is for a Maritime Patrol Aircraft (MPA) to detect and locate a submarine through the deployment of sonobuoys. As
such this is a search and track mission which is deemed to be complete when a stable track is established.

Intelligence information provides the MPA crew with the target type and an indication of the submarine's location and, hence, the area to be searched. Sonobuoys are dropped in one of a set of patterns dependent upon factors such as the speed and direction of the target, presence and strength of previous detections, the type of the crew and its workload. The particular sonobuoy pattern used is the decision of the TacNav. A procedure manual of tactics regarding what action to take in any situation is used and a newly qualified TacNav will closely follow these rules. However, as he becomes more experienced he will start to apply his experience and adapt the rules to better fit the situation. Hence, there is no well-specified algorithm to determine what pattern of sonobuoys should be deployed in any given set of circumstances.

2.1 Fusion in the ASW Application

Anti-submarine warfare is a task that includes two levels of manual fusion. The first level takes place when the sonar operators declare a target detection based on information provided by a set of sonobuoys previously deployed. The second level of manual fusion is carried out by the TacNav who uses the qualitative detection information provided by the sonar operators with track data and the perceived crew workload to make decisions about the next deployment of sonobuoys. The results of this latter fusion process may be inconsistent and dependent on the experience of the TacNav employed at the time.

The objective of this work was to combine the encapsulated experience of the TacNav with the rules provided by the tactics manual to provide a consistent advice tool.

2.2 The Key ASW Modelling Issue

The effectiveness of the system described above depends on the accuracy of modelling the TacNav's decision-making processes. Although the tactics regarding sonobuoy deployment are specified (for most circumstances) in a tactics manual, it has been found that the exact decision made will vary both between and within TacNavs. Thus the problem is one of modelling, not only the simple heuristics, but also the imprecise knowledge encapsulated in the mind of the experienced TacNav.

Various issues were identified, including the fidelity of the individual rules, the representation scheme used and the software implementation approach. It quickly became clear that a methodical approach to engineering the model as a whole was more important than the optimisation of these individual components.

The knowledge held in the TacNav's mind is a valuable asset and can be utilised in a disparate range of areas such as sonobuoy-use reduction, personnel training and mission optimisation. The TacNav provided judgmental information on how these decisions would be affected by external factors such as whether the crew was aggressive, whether or not the target had already been detected, the workload of the crew, etc. Such knowledge is not specified in the tactics manual and is only gained through experience.

Other knowledge is held in the tactics manual and records of previous missions. This makes it too disparate to be directly useful for our purposes. All of this knowledge needed to be collated and represented in a way that could be exploited to our advantage. It was felt that this stage would be usefully separated from the actual implementation.

The foregoing issues can be dealt with by KE, which is discussed in the next section.

3. An Overview of Knowledge Engineering

Knowledge within any organisation is commonly scattered between a number of personnel, documents and/or computer systems that may not even be located at the same site. Knowledge acquisition is the process of extracting knowledge from an expert. KE, of which knowledge acquisition is a component, focuses on the acquisition, modelling and management of this distributed fundamental domain knowledge, as well as any personal expertise.

KE covers a range of techniques including mathematical modelling, neural networks, genetic algorithms, knowledge-based and expert systems, data mining, natural language processing, intelligent agents, virtual reality, data visualisation and case-based reasoning. Expert systems are considered particularly beneficial.

As with anything else, there are advantages and disadvantages with KE. Disadvantages include a mistrust of the concept and hence little acceptance of the techniques. This is probably due to the fact that there is no proven track record in the field and that it appears to take a long time to develop anything usable. Another disadvantage is that there are very few knowledge acquisition and knowledge engineering
tools available. This means that development is usually conducted in a hybrid manner. On the other hand, advantages include an increased understanding of the processes under consideration at the end of the KE experience by both the expert and the knowledge engineer and a more robustly optimised process. Feigenbaum [2] summarises the three main advantages of KE as cost reduction, automated information processing and the gaining of new knowledge.

3.1 Approaches to Knowledge Engineering

The individual with the responsibility for collecting and structuring this knowledge and for developing the model with which to fuse it all is known as the knowledge engineer. A knowledge engineer has to elicit and manage large amounts of information-rich but ill-structured expertise data and needs a structured approach to help in this process.

A pre-requisite to any form of structured approach is a multi-disciplinary team. This has the advantage that a wider view of the knowledge available is obtained than if a single person is involved in the task.

The major tools for knowledge acquisition include interviewing, data analysis, text analysis, behaviour analysis and machine induction, the first two being the most popular. It is rare for one technique alone to be used in any knowledge acquisition task [3]. Some of these are briefly described below.

- Interviewing (learning by being told) provides information directly from the people with the knowledge and involves the knowledge engineer in studying verbal exchange, questionnaire responses, etc;
- Data Analysis is knowledge acquisition through analysing historical data records;
- Text Analysis is knowledge acquisition through the use of books, manuals, the internet, etc. It is a little used method but has the advantage that access to a busy expert is not necessary;
- Behaviour Analysis (also known as learning by observation) involves the knowledge engineer observing the expert in action and the expert justifying his actions;
- Machine Induction theoretically speeds up the process by collecting information in the form of case studies. A computer extracts the appropriate information to produce the required knowledge.

Winston [4] defines the basic questions to be posed regarding knowledge as:
1. What kind of knowledge is involved?
2. How should the knowledge be represented?
3. How much knowledge is required?
4. What exactly is the knowledge needed?

3.2 Methodologies for Knowledge Engineering

It is often difficult to go directly from the elicited knowledge to an implemented system. One reason for this is the confounding of different types of knowledge, i.e. task knowledge and domain knowledge, making it unclear how the system ought to be developed.

There are prescribed methodologies for KE, the right one to use at any one time depends on the situation. A generic KE life cycle appropriate for predictable systems, with rigid specifications that allow fixed price development and a disciplined manner of progression, includes:
- feasibility study including assessing the scope of the system, determining which parts of the system should be knowledge engineered and which parts should be conventionally programmed, which techniques to use, software integration issues, determining data and information availability, appraising cultural issues and identifying appropriate experts;
- requirement specification including defining and validating the knowledge, data representation and maintenance requirements, agreeing the users expectation of the system and how they wish to interact with it, determining mandatory and desirable requirements and producing performance specifications;
- system design;
- module design;
- module coding;
- module integration;
- acceptance testing requiring the availability of test data sets. This also covers the problem of how to validate knowledge and how to test safety critical systems. (Incremental testing could help in overcoming some of these problems.) Acceptance testing requires awareness of the original scope of the problems and identification of the quality of the tests being carried out;
- commissioning is similar to testing with the added problems of resistance to new technologies by the intended users. Commissioning requires feedback from users to assess the implementation and expectation issues.

Each stage should be formally documented and signed off before proceeding to the next stage. This gives rise to extra administrative costs and additional time if it is decided at a later date that earlier stages need altering.
For flexibly specified systems with vague or uncertain requirements and outcomes, the above life cycle needs to be moderated accordingly, but feedback should be tightly monitored. Examples of tools to use with these less formal methods include Rapid Application Development (RAD) [5] and Dynamic Systems Development Method (DSDM) [6], an implementation of RAD. These are general iterative prototyping approaches to software development. The idea of achieving a certain level of functionality within a fixed time period is not new, but these tools to facilitate it are.

### 3.2.1 CommonKADS

The two methodologies described above place the implementation of the acquired knowledge at the centre of the design process. This often leads to bespoke systems with little reuse of existing knowledge processing modules. During the last decade there has been a move away from this implementation-centric view to a knowledge-centric view in which the knowledge model and its implementation are maintained as separate entities during the design phase. CommonKADS is such a methodology, which is currently finding favour in European KE applications. It is a results-oriented methodology for developing a Knowledge-Based System (KBS) from application selection to design and testing. It is derived from KADS [7,8] that was developed during European Union funded ESPRIT projects (Projects 1098 and 5248) that ran between 1983 and 1994. The work was extended to develop KADS to become a European standard in the form of CommonKADS [9]. KADS is now widely used within European Union countries as a practical KBS development methodology.

The use of CommonKADS to develop a KBS is fundamentally a process of multi-perspective modelling. To this end CommonKADS provides a framework of representations and process suggestions for producing system descriptions at different levels of abstraction through the use of diagrams, text and / or graphical notations. These diagrammatic representations are considered to be the most useful parts of the approach.

The methodology can be split into three main components as shown in Figure 1 - the feasibility study, knowledge modelling and design and implementation.

The feasibility study comprises the production of:
- an organisational model which models the organisational environment in which the system will operate;
- a task model which describes, at an abstract level, the tasks which are necessary to realise some function within the organisation;
- an agent model which models the capabilities of the people and / or the computer systems that perform the tasks identified above.

The knowledge modelling comprises the production of:
- a communications model that models the communications among the agents involved in a task. The purpose of this model is to identify some of the risks associated with the user interface;
- an expertise model which models the problem solving capability of the agents involved in the task. The knowledge required in this model can be separated into three types:
  - *domain knowledge* which is knowledge about the physical and conceptual systems being tackled;
  - *inference knowledge* which describes the inferences that can be made using the domain knowledge;
  - *task knowledge* which specifies the goals and activities making up the task and the order in which the inferences will be used.

The design and implementation phase includes:
- the design model which describes the structures and mechanisms of the systems which are involved in the task.

The CommonKADS methodology facilitates a library of re-usable models or part-models for frequently used types of task.
4. Developing the Anti-Submarine Warfare Model

A multidisciplinary team was available to work on the tool. This included mathematicians, staff familiar with different aspects of the ASW application and an experienced TacNav.

Despite its widespread use in the KBS community, there was no evidence that CommonKADS had been applied to a data fusion system. It was decided that CommonKADS would provide a valuable development tool in many data fusion applications and that the entire ASW TacNav aid development could be addressed using the CommonKADS methodology. The results are shown in Figures 2-7, although it should be noted that for classification reasons, the models shown might not always be complete.

4.1 The ASW Feasibility Study

The Organisational Model established a basic organisational context within which the TacNav aid would operate. This is shown in Figure 2.

![Mission Commander](image1.png)

The Agent Model identified the main human and computer elements present in the organisation and detailed their capabilities as shown in Figure 3. From this, it was decided to ignore the existence of the radar and radar operator in the initial computer model.

![Agent Model for ASW](image2.png)

The Task Model identified and related the different tasks (excluding radar) performed during an ASW mission. This is shown in Figure 4.

![Task Model for ASW](image3.png)

<table>
<thead>
<tr>
<th>AGENT</th>
<th>CAPABILITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sonobuoys</td>
<td>able to passively detect sub-surface targets and to provide intensity and Doppler information</td>
</tr>
<tr>
<td>Radars</td>
<td>able to detect surface targets (omitted from initial model)</td>
</tr>
<tr>
<td>Sonar operators</td>
<td>able to assess the sonobuoy information in context to call target contacts and to judge their confidence</td>
</tr>
<tr>
<td>Radar operators</td>
<td>able to assess radar information in context to call target contacts (omitted from initial model)</td>
</tr>
<tr>
<td>Tracker</td>
<td>able to maintain an estimate of target location using sonar derived bearing estimates</td>
</tr>
<tr>
<td>TacNav</td>
<td>able to assimilate the information from the above and other sources</td>
</tr>
</tbody>
</table>

![Communications Model for ASW](image4.png)
4.2 The ASW Knowledge Modelling

The Communications Model identifies what information is passed, where it comes from and where it is passed. This is shown in Figure 5.

The Expertise Model is separated into three (Figures 6a-c) and identifies the knowledge relating to the ASW problem domain, the conclusions that can be reached using the domain knowledge and the activities to be carried out and the order in which this is done.

4.3 The ASW Design and Implementation

The knowledge models were used to design a block structure for the decision aid that was connected using information derived from the communications model. The assignment of functions to particular software modules was done with reference to the organisation, agent and task models.

4.3.1. The ASW Design

A computer model of the ASW scenario was developed. This model was greatly simplified by making some assumptions including:

- there is only one target being limited in course and speed and whose action is not affected by that of the MPA;
- there is only one aircraft searching with a fixed maximum number of sonobuoys being deployed at any one time;
- the sonar operators as a group and the Tactical Navigator are of average ability;
- all crew members are aware of the area of interest and target type.

Figure 7. The Design Model for ASW
The model comprised three components illustrated in Figure 7:

- a decision simulator to simulate the TacNav's decision-making process;
- a data simulator to provide time-varying parameters, such as submarine position, of the external environment;
- a set of interfaces to provide linkages between different parts of the model.

### 4.3.2. The ASW Implementation

It was decided to implement the design shown in Figure 7 in two parts. The data simulator and the interfaces were written in C++ using standard software engineering practices. The decision simulator was implemented using a Bayesian Belief Network (BBN).

The purpose of the decision simulator in the ASW computer model was to predict the next action to be taken by the TacNav. The TacNav makes his decision by fusing a variety of data and information, some of which is uncertain, and then evaluating all of his options to produce a set of actions each of which is associated with a likelihood. Work at DERA has previously shown that complex military applications can be modeled using BBNs [10]. Since these allow incorporation of uncertainty into the model and produce an uncertain output, the use of a BBN for this application was considered appropriate.

A Bayesian Belief Network comprises nodes and directional links that depict the relationship and dependencies between uncertain data. Nodes may have parent nodes and child nodes. A parent node is one whose value affects a child node. An example of a simple BBN is shown in Figure 8 where the value of target_speed depends on the values of target_type and target_position, and hence target_type and target_position are the parent nodes of target_speed. Similarly, target_speed is a child node of both of target_type and target_position.

Each node may assume one of a number of states. For example, the target_speed node Figure 8 could take the values fast, medium or slow.

BBNs allow information about uncertainties associated with any node to be propagated through the network and the uncertainties of parent and/or child nodes to be updated based on this new information. So if target_speed is slow, but a new measurement has just been made which shows the new target position to be a long way from the previous target position (assuming periodic measurements), then target_speed can be updated to medium.

Each node has associated with it a set of conditional probabilities, known as the Conditional Probability Matrix (CPM). This indicates the probability of each state of the node given all combinations of the parent node states. The default values of the CPMs of nodes without parents are the prior probabilities of the states. When something happens to change these prior probabilities, this change is propagated through the BBN updating all subsequent CPMs using probability theory. One problem with the CPMs is size. If, in the above example, there are only two target types (A and B), three target positions (same, close and far) and three target speeds (slow, medium and fast) it can be seen that for even such a small network, the CPM for the node target_speed is large. In cases where a node has more than two parent nodes and/or any of the nodes have many states, the size-problem can become unmanageable.

A BBN can also work backwards. In the example above we could ask “Given that the target_speed is fast, what is the probability that the target is of type A?” This can be found using Bayes theorem.

The commercial package HUGIN [11] was used to implement the ASW BBN. This was chosen because previous work had indicated that it was suitable for the purpose as well as being readily available, able to run on a PC and operable from within a C++ program using an application programming interface (API).

The BBN developed for this application was only used for forward propagation, although there is no reason why it could not be used for backwards propagation as well to perhaps assess the performance of other components of the model.
The decision simulator itself was split into sub-components:

- the crew environment which models the estimated crew work-load;
- the sonar operators which are modeled as a group rather than individually. This sub-component fuses the detection and signal strength information from the sonobuoys to provide qualitative detection and strength estimates for target contact;
- the tactical navigator which fuses contact information from the sonar operators and the estimate of the crew work-load to produce a tactical decision.

The completed implementation has subsequently been tested by domain experts and is currently being considered for further development. Full details of the whole computer model can be found in [1].

5. Conclusions

The authors had previously taken an algorithmic approach to data fusion system development, and regarded the inclusion of judgmental information as outside their domain. In developing this data fusion system it became clear that the problem of including judgmental information had to be addressed. After some unstructured preliminary attempts, it became clear that a methodical approach needed to be followed. We would recommend the use of a sound knowledge engineering methodology in such cases. We found CommonKADS to be a useful, albeit somewhat unwieldy, approach.

Our difficulties in using CommonKADS included:
- the representation of the knowledge was different at the different layers;
- the diagrams could not make recursive processes explicit;
- even a small system produced a large quantity of documentation.

Advantages we have observed in using CommonKADS included:
- the solution was captured irrespective of the final implementation;
- the specification of system functionality was (properly) documented;
- the different conceptual types of knowledge were appropriately distinguished making the final model easier to understand;
- it seemed that large systems would be more easily maintained.

References

[1] Unpublished DERA / MoD data


