Implementing C4ISR Architecture Framework -- An Australian Case Study

Paul Prekop and Gina Kingston
Joint Systems Branch
Defence Science and Technology Organisation
DSTO Fern Hill, Department of Defence, Canberra, ACT, 2600
AUSTRALIA

Abstract

Architectures are an emerging approach for capturing complex knowledge about organisations and systems. Architectural approaches range from broad, enterprise focused approaches, through to approaches aimed at specific user communities. Within defence community, the US DoD's C4ISR Architecture Framework (C4ISR AF) is emerging as one method for capturing the knowledge of how a defence force can be organised for particular situations.

This paper describes the process undertaken for the development of a case study of the C4ISR AF, the Joint Operational Architecture Demonstrator. The paper includes recommendations on the product construction sequence and the implications for the dependencies between the various C4ISR AF products. It also describes the development of tool support, in the form of customised extensions to PTech Inc.'s Framework, used by the team to develop the C4ISR AF products.

1 INTRODUCTION

The C4ISR AF has become a defacto-standard for the development of architectures with the Defence community. However, there are few papers that describe the practical application of the framework. This paper attempts to bridge the gap between theory and practice by discussing lessons learnt during the application of the C4ISR AF to the development of an architecture for a military operation.

This paper describes some aspects of the process used to construct the Joint Operational Architecture Demonstrator (JOAD), which used C4ISR AF to represent an operation. The approach presented in this paper strongly correlates with process developed by Levis and Wagenhals (2000). The paper focuses mainly on practical lessons learned, and insights gained while applying the process.

1.1 The Joint Operational Architecture Demonstrator

The Joint Operational Architecture Demonstrator (JOAD), undertaken by staff within DSTO's Joint Systems Branch (JSB), used version 2.0 of the C4ISR AF to represent how the Australian Defence Force (ADF) would be deployed to respond to a Defence of Australia (DAA) scenario. The DAA scenario represents, potentially, the largest callout of Australian forces, and the most complex configuration of Australian assets under warfighting conditions.

The scenario, the Operational Concept, indicative tasking, and ORBAT, which were used as the basis for JOAD, were taken from a Force Options Testing wargame run jointly by the ADF's Military Strategy Branch and DSTO's Theatre Operations Branch. The wargame was a seminar style wargame, and involved senior offices (LtCol and above) from various branches of the ADF. In the wargame, a scenario, together with an ORBAT was presented to the wargame players, who developed a high-level plan for the operation.

As the development of JOAD progressed, the initial data obtained from the wargame was augmented with information and expertise gathered from senior staff within the ADF and records from previous operations. The resulting data represents a detailed, hypothetical, description of how the ADF would react in a DAA scenario.

The focus of the JOAD was primarily on the C4ISR AF's Operational View; the collection of products that describe the warfighting context, the organisation of the warfighting elements, and the information flows between the different elements. Six types of OV products[1] were developed:

- OV-1 High Level Operational Concept Graphic;
- OV-2 Operational Node Connectivity Description;
- OV-3 Operational Information Exchange Matrix;
- OV-4 Command Relationship Chart;
- OV-5 Activity Model, and
- OV-7 Logical Data Model.

As well as the OV products, JOAD also included the two essential context products the AV-1 Overview and Summary Information, and the AV-2 Integrated Dictionary. JOAD also includes a new product type, the OV-X Operational Node to Activity Relationship Diagram (see Section 2.3.2). Due to resource constraints, the systems views were not investigated.

The effort required to develop JOAD was considerable, and involved three staff over a nine-month period. The final demonstrator contains 3115 objects (of 14 different types) spread over 124 models and diagrams that are used to make up the nine different C4ISR AF products developed. The development effort can be roughly divided into four main stages:

- Collecting data, including elaboration of the scenario and the mission;
- Entering and importing data into the architecture tool, and constructing the C4ISR AF products;
- Enhancing and modifying the architecture tool, and
- Exporting the C4ISR AF products to HTML for WWW publication.

The following graph shows how the staff effort was split over the nine months.

[1] Since the resulting architecture was classified, actual examples of the JOAD architecture products can't be included in this paper.

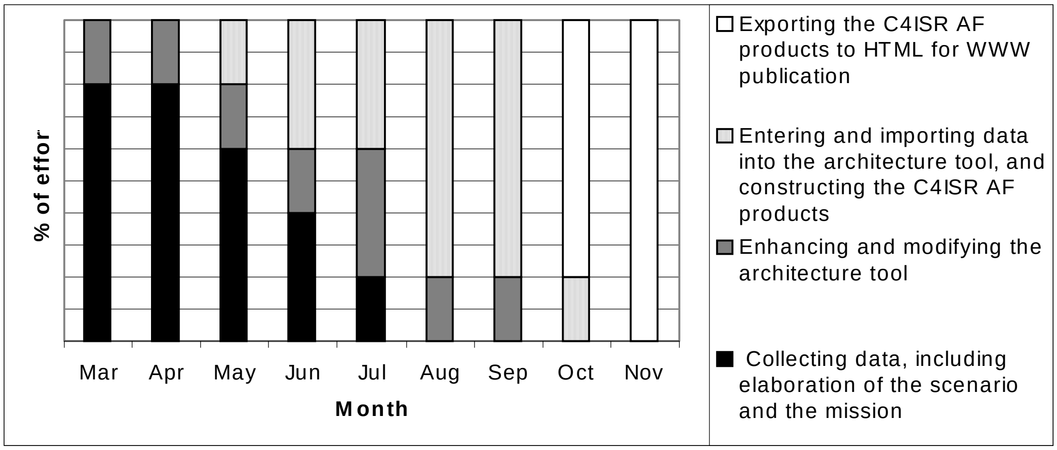

Figure 1: JOAD development effort by stage.

As the above graph shows, over one third of the JOAD development activity was spent on data collection. Entering and importing data into the development tool and developing the C4ISR AF products also took about a third of the overall development time.

The remaining third of the development time was divided between enhancing and modifying the architecture tool, and exporting the products to HTML for WWW publication (Enhancing and modifying the architecture tool is described in more detail in Section 3).

2 C4ISR AF DEVELOPMENT APPROACH

As pointed out by Levis and Wagenhals (2000) there is a strong dependency between the architecture products; some supporting products need to be built before some essential products. The order of the products as they appear in the C4ISR AF document is not an indication of the order in which they should be developed.

Based on the experiences of the team in the development of JOAD, we propose a C4ISR AF development approach, similar to that described by Levis and Wagenhals (2000), that consists of five key stages:

- Stage 1. Describe the context and goal of the architecture;
- Stage 2. Represent the activities to be performed;
- Stage 3. Represent the force elements involved;
- Stage 4. Represent information needs and information flows, and
- Stage 5. Create the integrated dictionary, finalise the architecture overview.

Many of the stages include several detailed steps as shown below in Table 1, with each stage producing one or more C4ISR AF products.

Stage 1. Describe the context and goal of the architecture.	
Step 1.1. Choose the purpose and scope of the architecture.	• AV-1 Overview and Summary Information (Architecture purpose).
Step 1.2. Describe the operational concept.	• OV-1 High Level Operational Concept Graphic.

Stage 2. Represent the activities to be performed.	• OV-5 Activity Model.
Stage 3. Represent the force elements involved.	
Step 3.1. Select the operational nodes; devise and represent the command hierarchy.	• OV-2 Operational Node Connectivity Description (Nodes Only), and • OV-4 Command Relationship Chart.
Step 3.2. Relate activities to operational nodes.	• OV-X Operational Node Activity to Relationship Diagram.
Stage 4. Represent information needs and information flows.	
Step 4.1. Identify information and develop the data model.	• OV-7 Logical Data Model.
Step 4.2. Identify and represent needlines and IERs.	• OV-2 Operational Node Connectivity Description (Needlines final), and • OV-3 Operational Information Exchange Matrix.
Stage 5. Create the integrated dictionary, and finalise the architecture products.	
Step 5.1. Finalise the architecture overview.	• AV-1 Overview and Summary Information (final).
Step 5.2. Finalise the architecture products.	• OV-1 High Level Operational Concept Graphic (final); • OV-2 Operational Node Connectivity Description (final); • OV-3 Operational Information Exchange Matrix (final); • OV-4 Command Relationship Chart (final); • OV-7 Logical Data Model (final), and • OV-X Operational Node Activity to Relationship Diagram (final).
Step 5.2. Create the integrated dictionary.	• AV-2 Integrated Dictionary.

Table 1: Relationships between development the steps and C4ISR AF products.

2.1 Stage 1. Describe the context and goal of the architecture

This stage of the development activity defines the purpose, scope and context for the architecture development effort, as well as defining the scenario and context of the operation to be captured by the architecture.

This stage is refined into two distinct steps. Step 1.1 produces the AV-1 Overview and Summary Information. Step 1.2, develops the OV-1 High Level Operational Concept Graphic.

2.1.1 Step 1.1 Choose the purpose and scope of the architecture

The AV-1 Overview and Summary Information used in JOAD captured the purpose, scope and context of the architecture. The AV-1 was implemented as a simple collection of text fields that captured the information. This step is well defined by the C4ISR AF document. During this stage, the development team also explored and assessed different architecture tools (architecture tools are described in Section 3).

2.1.2 Step 1.2 Describe the operational concept

The operational concept is the overarching description of the operation being represented by the architecture. The C4ISR AF captures the operational concept in the OV-1 High Level Operational Concept Graphic. However, the C4ISR AF isn't prescriptive on the format of an OV-1, suggesting a simple, conceptual graphic showing the operation's key entities and their relationships.

As pointed out by Levis and Wagenhals (2000), defining a good operational concept is difficult, because the operational concept must capture a detailed enough description of the operation to be useful during more detailed planning, while not constraining the more detailed planning of the operation.

The operational concept used with JOAD was taken from the Force Options Testing wargames as described previously in Section 1.1. We found it useful to capture the following information in our Operational Concept:

- Mission geography (the geographical layout of the mission elements);
- The mission objectives, as related to the elements of that geography, and
- Indicative force characteristics for each objective/geographic area combination.

We also found that a simple graphic wasn't enough to completely capture the Operational Concept, and found it useful to provide short narratives of what the graphics depicted. We also divide the operational concept into three stages, represented by three different, but related, OV-1s.

Within JOAD we also found it useful to provide a way of navigating from key operational nodes captured in the OV-1 to the more detailed operational node representations used in the OV-2 Operational Node Connectivity Description. This provided a useful way of moving from the context provided by the OV-1, to the detail provided by the OV-2. We also found it useful to link high-level mission objectives to specific activities. This provided a way of describing how the various mission objectives captured by the OV-1 were satisfied by the activities captured in the OV-5 Activity Model.

2.2 Stage 2. Represent the activities to be performed

This stage of the development activity captures the tasks or activities needed to satisfy the mission objectives identified in the OV-1 High Level Operational Concept Graphic. Once identified, the tasks or activities can then be captured in the C4ISR AF's formal OV-5 Activity Model. The C4ISR AF recommends using the IDEF0's ICOM activity model, although it doesn't preclude other modelling approaches from being used.

The OV-5 can be seen as a formalised representation of an essential task list. Processes for developing essential task lists, have been published by both the UK and US (DoD 1995; MoD 1999). Both of these approaches rely on some kind of Joint Task List (JTL) from which essential tasks are selected. Since Australia doesn't have a JTL, the development team drew tasks from the indicative tasks provided by the wargame, Australian Army Tactical Task List, the US Universal Joint Task List, as well as the UK Joint Essential Task List.

While the OV-5 isn't a mandatory product within version 2.0 of the C4ISR AF, we found it difficult, if not impossible, to construct a complete OV-2 Operational Node Connectivity Description and OV-3 Operational Information Exchange Matrix without the basic information that would make up the OV-5. If a formal OV-5 isn't required by the architecture, developing an essential task list should provide the information needed to complete the OV-2 and OV-3 products.

For JOAD, we developed our own activity modelling approach, based on the Action-Object-Location-Qualifier (A-O-L-Q) model, shown below by Figure 2. The A-O-L-Q model was derived from a grounded theory analysis of existing, UK, AS and US task lists.

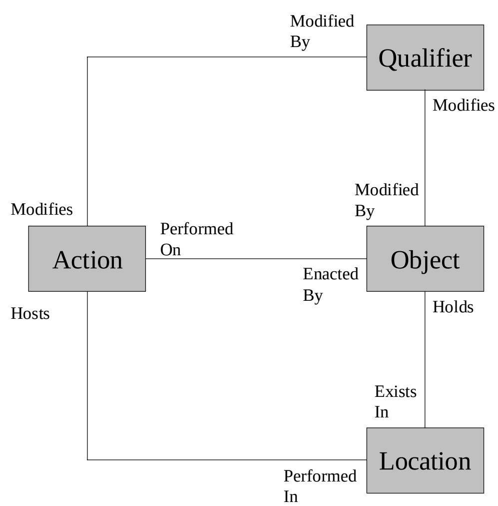

Figure 2: The A-O-L-Q model.

Within the A-O-L-Q model, each activity is represented by four related components. Actions describe what is done, or what will be done by the activity, for example, *Develop* or *Patrol* and so on. Actions are performed on an object, or subject of the activity. For example, in the activity *Develop Theatre Wide C2 Plan*, *Develop* is the action performed on the object *Theatre Wide C2 Plan*.

The qualifier and location are optional, and both refine the behaviour of the action. The qualifier modifies the action performed on the object. In some activities, this might be to limit, or better specify the action, for example in the activity *Develop Theatre Wide C2 Plan for an International Operation* the qualifier *for an International Operation*, modifies the behaviour of the *Develop* action on the *C2 Plan* object. Location confines the scope of an activity to an area. Location may be a physical area, for example *Northern Australia*, it may be a logical area, for example, the *Air-Sea Gap*, or it may be an environment, *Land Environment*, or *Cyber Environment*, or even a conceptual area, for example *International Community*.

2.3 Stage 3. Represent the force elements involved

This stage of development identifies and describes the operational nodes or physical force elements that will conduct the activities identified in Stage 2.

Step 3.1 describes the identification of operational nodes, and the production of the initial OV-2 Operational Node Connectivity Description, and OV-4 Command Relationship Chart. Step 3.2 describes the production of the OV-X Operational Node to Activity Relationship Diagram.

2.3.1 Step 3.1 Select the operational nodes; devise and represent the command hierarchy

The C4ISR AF OV-2 Operational Node Connectivity Description captures the operational nodes, while the OV-4 Command Relationship Chart captures the related command hierarchy.

In determining the operational nodes for the OV-2, we considered the capability required at the key locations within the operation, the mission objectives (both were captured in the OV-1 High Level Operational Concept Graphic), and the activities that needed to be performed (which were captured in the OV-5 Activity Model). These capability requirements naturally formed logical operational nodes. Each logical node was then mapped onto a collection of physical force elements that would provide the capability required.

Once the operational nodes were identified, the initial OV-4 Command Relationship Chart was developed. The nature of the command structure used within the operation was partly dependent on the operational nodes that exist, the activities that needed to be performed, as well as on the command concepts and mission structure given by the OV-1. The development of the OV-4 resulted in the identification of additional operational nodes.

The C4ISR AF provides little guidance on how to structure an OV-4. As shown in Figure 3, we found it useful to structure our OV-4 product around the concept of a command position. Each operational node (the oval) has an associated commander (the person icon), who was represented in a command hierarchy. We chose this approach because we found it important to focus on command responsibilities within the commander and not with an operational node.

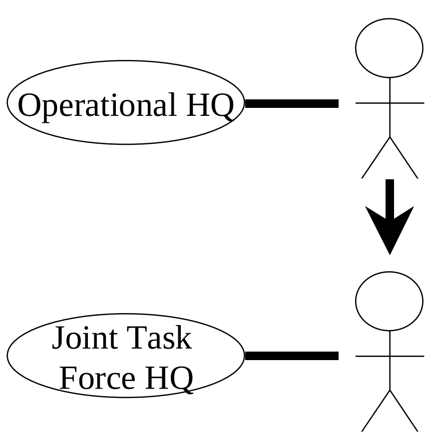

Figure 3: A Simple OV-4 Command Relationship Chart.

When developing the OV-2 and OV-4 products, we found a single diagram unwieldy because of the large scale of the operation we were representing. We constructed multiple OV-2 and OV-4 diagrams structured according to the command hierarchy. Where this was still unwieldy, we further divided these diagrams into separate diagrams for each of the functional areas in the OV-5.

2.3.2 Step 3.2. Relate activities to operational nodes

The C4ISR AF captures the relationships between operational nodes and the activities they perform as part of the OV-2 Operational Node Connectivity Description and OV-3 Operational Information Exchange Matrix. However, the relationships between operational nodes and activities should be identified before information needlines and information exchange requirements are added, because the relationship can act as useful information predictor. To explicitly capture these relationships, we developed an additional product, the OV-X Operational Node to Activity Relationship Diagram.

The OV-X can be as simple or a complex as needed. In its simplest form, the OV-X only captures the *Node Performs an Activity* relationship. However, more complex relationships between operational nodes and activities could also be included. A simple OV-X is shown below in Figure 4. This diagram captures a simple OV-X showing the relationship between operational nodes (represented by the oval) and activities (represented by the circular arrows).

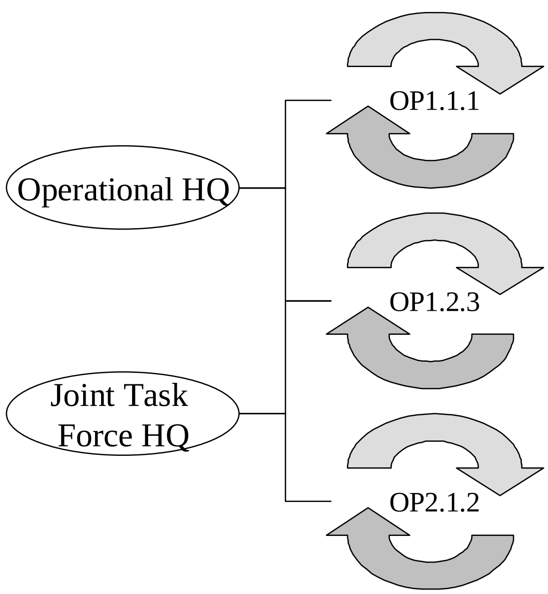

Figure 4: A Simple OV-X Operational Node to Activity Diagram.

For JOAD, we found that we could produce an initial, simple, OV-X from the information produced in the previous stages. The OV-1 High Level Operational Concept Graphic, provided indicative relationships, while the OV-5 Activity Model and initial OV-2 provided details of the activities and operational nodes. The OV-4 Command Relationships Chart provided further information on C2 relationships.

2.4 Stage 4. Represent information needs and information flows

The next stage in developing an architecture for an operation is to establish the information needs and information flows, captured as the needlines between the operational nodes. This stage may also identify crucial information processing nodes omitted from previous steps, and may help to determine the system requirements, although system requirements were not developed for the JOAD.

Step 4.1 develops the OV-7 Logical Data Model. Step 4.2 describes the creation of the OV-2 Operational Node Connectivity Description and the OV-3 Operational Information Exchange Matrix.

2.4.1 Step 4.1 Identify information and develop the data model

The C4ISR AF's data model, the OV-7 Logical Data Model, is a supporting product within version 2.0 of the Framework. The C4ISR AF is not prescriptive about what details the OV-7 should contain, or how it should be structured. The role of the OV-7 is to describe the information that is passed within the needlines as a part of the Information Exchange Requirements (IER) captured in the OV-3 Operational Information Exchange Matrix.

For JOAD we developed a simple, high-level information taxonomy to capture the information elements that were likely to be exchanged within the operation being represented by the architecture. The taxonomy was developed by analysing existing ADF operational information. During the analysis, information content, and information structure emerged as the two most useful and universal characteristics of information. The taxonomy codified operational information along two axes, the Information Content and the Information Structure. Specific Information Elements are then described as the intersection of these two axes. Figure 5, below, shows an information model for environmental information.

Information Content	Summaries	Maps/Charts	Report	Alert/Notice	Assessments
National Environments	E1				
Atmospheric Environment		E2			
Physical Environment					
Natural Physical Environmen					
Health Environment				E2	
	Information Structure				

Figure 5: Example Information Model for Environmental Information.

In the example above, the information elements (E1, E2, and E3) are defined by the intersection of the information content (capturing environmental information) with the information structure (capturing structures relevant to environment information).

2.4.2 Step 4.2. Identify and represent needlines and IERs

The C4ISR AF uses the OV-2 Operational Node Connectivity Descriptions to capture the information flows, or needlines, between operational nodes. The OV-3 Operational Information Exchange Matrix is used to capture the detail of the needlines in the Information Exchange Requirements (IERs).

The development of OV-2 and OV-3 products for JOAD began by identifying the information flows required by each functional area identified in the OV-5 Activity Model. The OV-4 Command Relationship Chart was used to guide the identification of Command and Control information flows. The OV-X Operational Node to Activity Relationship Diagram then guided the identification of the details of the IERs associated with a needline.

Depending on the type of tool support being used to develop the architecture products, the IER information can be added as the needlines are added to the OV-2 product, or as we did in the JOAD development, the IER information can be included after the needlines have been added to the OV-2 product.

2.5 Stage 5. Create the integrated dictionary and finalise the architecture products

The final stage in the development of the architecture is to finalise the AV-1 Overview and Summary Information, to ensure that all of the products are consistent, and to generate the AV-2 Integrated Dictionary, which describes all of the elements in the architecture.

Step 5.1 produces the final AV-1 product. Step 5.2 ensures the consistence of the products. Step 5.3 produces the final AV-2.

2.5.1 Step 5.1 Finalise the architecture overview

As described in Step 1.1, the AV-1 Overview and Summary Information captures the specific purpose, scope and context of the architecture being developed. The AV-1 also captured details of the products developed, the results of any analysis performed on the products, the tools used to generate the products, file formats and locations used to store the products developed, and so on.

In the JOAD development we finalised the AV-1 product by adding descriptions of the architecture products developed, tool support used, product locations and authorship.

2.5.2 Step 5.2 Finalise the architecture products

Toward the end of the architecture development effort, it is important to revisit all the architecture products developed to ensure consistency between the products, and between the products and the operation they capture. The earlier in the development a product was created, the more likely it is to require changes.

After the needlines and IER had been added to the OV-2 Operational Node Connectivity Description and OV-3 Operational Information Exchange Matrix, we identified new information centric operational nodes and command relationships, and information centric activities that were not identified earlier. The OV-2, OV-4 Command Relationship Chart, and the OV-X Operational Node to Activity Relationship Diagram were then reviewed and modified.

2.5.3 Step 5.3. Create the integrated dictionary

The final C4ISR AF product to develop is the AV-2 Integrated Dictionary. Depending on the types of tools used to develop the architecture products, it may be possible to machine generate the AV-2 based on the information held in each of the C4ISR AF products. We successfully used this approach during the development of the JOAD.

However, if tool support isn't available to generate the AV-2, based on the developed products, it may be worth developing the dictionary as each of the products is developed, rather than leaving it to the end of the development activity.

3 TOOL SUPPORT FOR C4ISR AF DEVELOPMENT

The JOAD development effort used PTech Inc.'s Framework to support the development of the C4ISR AF products. Section 3.1 discusses the use, and extensions made to Framework, and Section 3.2 discusses the benefits of using tool support.

3.1 Tool Support and Development

Framework is a comprehensive, extendible, architecture tool, which provides support for the development C4ISR AF products through the use of an extension to Framework, the Military Information Architecture Accelerator (MIAA)[2].

The version of the MIAA used for the JOAD development effort was an early release, which provided support for the AV-2 Integrated Dictionary, OV-1 High Level Operational Concept Graphic, OV-2 Operational Node Connectivity Description, OV-3 Operational Information Exchange Matrix, and OV-4 Command Relationship Chart, as well as the SV-1 Systems Interface Description, SV-2 Systems Communication Description, and SV-3 Systems[2] Matrix.

However, this version of the MIAA didn't support all the products required for the JOAD development. The MIAA also had a US centric perspective, and enforced some interpretations of the C4ISR AF that differed from those of the development team. Consequently, the development team enhanced the MIAA to include support for the AV-1 Overview and Summary Information, OV-5 Activity Model, and OV-7 Logical Data Model, as well as the new OV-X Operational Node to Activity Relationship Diagram. The development team also customised the support for existing products, adding additional elements, relationships and attributes, to support ADF terminology and structures.

[2] A detailed review of Framework, within the context of operational architectures can be found in (Kingston, Prekop et al. 2000). A review of Framework within the context of general architecture tools can be found in (Prekop, Kingston et al. 2001).

Modifying and enhancing the MIAA is generally a simple process. The structure of the C4ISR AF products are captured in the MIAA by metamodels, and supported by set-based queries, text-based interfaces to the repository, called forms, and reports that can process the information in the repository. Framework supports the creation of new metamodels, queries, forms and reports as well as their extension.

Metamodels are literately models about models. They describe what objects exist within a product, what relationships exist between the various objects within the product, and the relationships between the different products. The development team added metamodels to support new products as well as extensively modified many of the existing MIAA metamodels.

Figure 6 and 7 are examples of a metamodel and a form added by the development team. The metamodel and form support the OV-5 Activity Model discussed in Section 2.2.

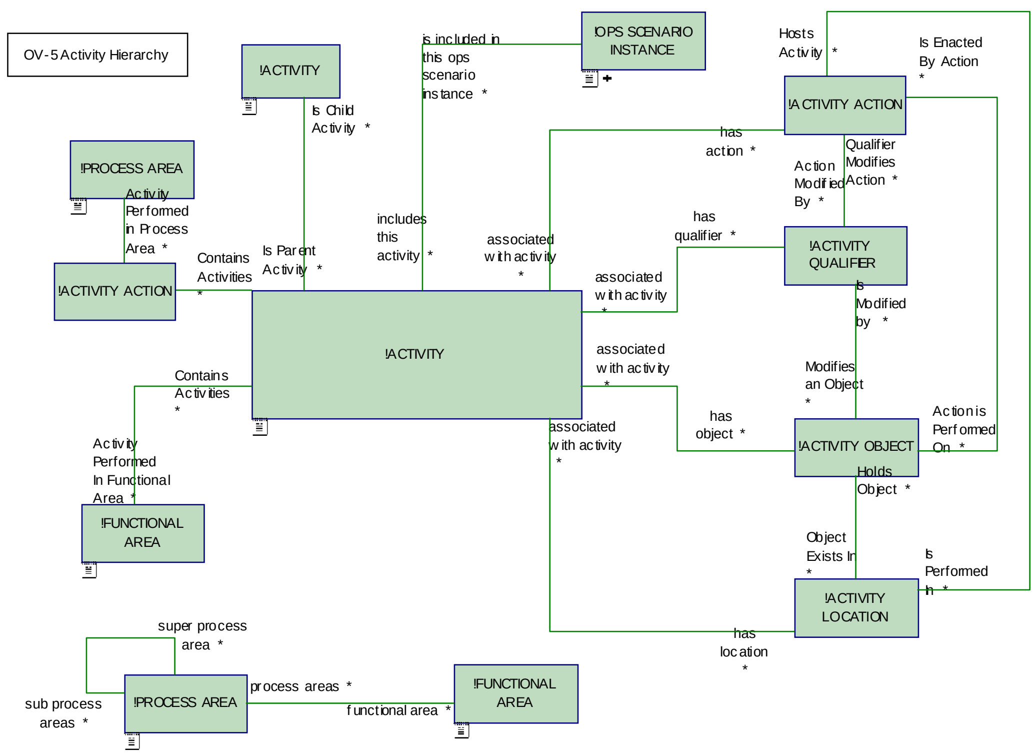

Figure 6: Metamodels to support extensions to the activity model.

The A-O-L-Q model shown by Figure 2 is captured in the right side of the metamodel in Figure 6. The metamodel also links the A-O-L-Q model to the pre-existing structures defined in the MIAA. The form shown by Figure 7, is used to capture additional activity information into the Framework repository.

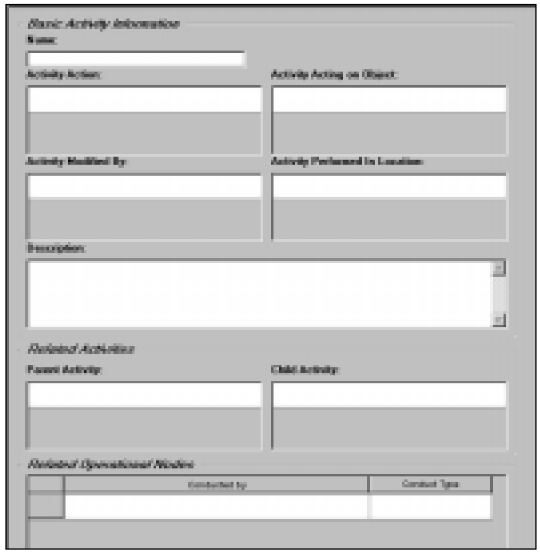

Figure 7: Activity Model Form.

Forms are used extensively within the MIAA to display additional textual information held for a product, object or relationship within or between products. Forms can also be used to display the results of queries. The development team developed forms for many of the products and objects added, as well as modifying existing forms to reflect the changes made to the metamodels.

Queries can be associated with particular products, objects or relationships within a product, and can be used to access almost all the data held within the repository. The results of a query can be used as the data on a form. Framework and the MIAA are supplied with a large collection of default generic queries. The development team added several new queries to check and manage information exchange requirements.

Framework's general purpose report system allows almost any data held in the repository to be extracted, and written out to external files in a text-based structure. Reports are written using a proprietary, stack based, reporting and formatting language called the Template Language. Reports are extensively used by Framework to support its own code generation functions.

The MIAA uses reports to generate the AV-1, AV-2, and OV-3. The development team created new template files to generate the AV-1 product, as well as modifying the template files that generated the AV-2 and OV-3 products.

3.2 Benefits and Limitations of Tool Support

Framework and the MIAA provided good support for the JOAD development effort. One of the key features provided by Framework was the ability to easily re-use data between the different products. For example, operational nodes entered into an OV-2, could be re-used when the OV-4, OV-X, and OV-3 products were developed. This helped to speed up the development process, because the data only needed to be entered once, and helped to ensure consistency between the products. Framework also provided support for completeness checking, navigation through the architecture, and incorporating data from other sources.

Framework also provided some support for enforcing the modelling rules for each C4ISR AF product. For example, it prevents illegal relationships between objects from being added. However, it doesn't prevent illegal objects from being added to products. Another useful feature, and one used extensively during the development of the OV-2 and OV-3 products, is Frameworks ability to perform consistency and completeness checking, highlighting missed needlines, or needlines without IER information and so on.

The use of tools is not without limitations. Framework imposed a learning curve on the development team. While all members of the team were familiar with architecture development and conceptual modelling, none had used Framework before. While basic Framework features were easy to grasp, the more complex features took some time to master.

Another problem encountered while using Framework is the lack of flexibility provided. The C4ISR AF is often very flexible in its descriptions of what constitutes the various C4ISR AF products. However, Framework only implements a fixed and rigid interpretation of C4ISR AF, that often differed from the development team's interpretations.

A useful feature not fully explored by the development team, is the architecture analysis functions provided by the Framework. This could be performed via a combination of Forms, Queries and Reports.

4 LESSONS LEARNT AND CONCLUSIONS

The first and most important lesson learnt as a result of the JOAD development effort, is the tight dependence between many of the C4ISR AF products. Many supporting products must be developed before the essential products are developed. The order of the products in the C4ISR AF document is not an indication of the order in which they need to be developed. This correlates well with the experience of Levis and Wagenhals (2000). A related lesson was the importance of the command hierarchy and mapping of activities to operational nodes, for identifying and describing information needlines and IERs.

Over one third of the JOAD development effort was devoted to identifying and collecting the raw information needed to develop the C4ISR AF products. One real difficulty encountered, was that much of the information needed to produce JOAD is generated by military planning activities, and as a result, the information doesn't exist, and won't be created until it is needed. Since the DAA scenario represented in JOAD is purely hypothetical, it became difficult to find information that described how the ADF would be deployed to deal with the scenario.

Managing the data that was collected by the development team also became a problem. The raw information was collected in both paper and electronic formats, and had a variety of security caveats. Even though the development team was small, and was collocated in the one building, finding information the team had already collected, and disseminating collected information within the team did become a problem. In hindsight, a flexible document management system may have addressed many of the information management problems the development team encountered.

The final information management problem the development team encountered was in the development of the actual C4ISR AF products. The final JOAD demonstrator is large, and as the number of individual diagrams and tables grew, it become increasingly important to structure the C4ISR AF products so they could be navigated and easily comprehended, by creating top level products which were linked to detailed, lower level products. However, our efforts were only partially successful, and JOAD is still difficult to navigate and comprehend. New methods, techniques, and tool support for managing and navigating the outputs of large architecture efforts are needed.

As discussed in Section 1.1 Framework was enhanced and modified throughout most of the JOAD development effort. As a need for new functionality was identified, or problems with the existing structures discovered, the tool was modified and enhanced. However, modifying and enhancing Framework in this way did result in us having to redo some products to accommodate changes made to the tool.

The development team achieved its goal of examining the utility of C4ISR AF for operational architectures. An added advantage of this work was gaining a greater understanding of the C4ISR AF, and the dependencies and relationships between the products.

Overall the C4ISR AF captures the information aspects of a military operation very well. However the C4ISR AF fails to capture many of the important relationships between products, especially between the OV-1 High Level Operational Concept Graphic and the rest of the architecture, and the relationships between operational nodes and activities. In its native form, the OV-1 doesn’t link back into any of the more detailed OV or SV products. This limits the utility of the OV-1 as a high level description of the operation because there isn’t a direct relationship between the OV-1 and the underlying details of the operation. As discussed previously, we modified the OV-1 to include relationships between the activities, and the operational nodes.

The C4ISR AF also fails to capture any notion of the flow of the operation, and the transition between different stages. We attempted to capture this by modifying the structure of the OV-1 product, to group the operation into different stages.

Currently the C4ISR AF doesn’t capture the independent relationship between activities and operational nodes. Within the C4ISR AF, this relationship is only described when mediated by information flows and captured in the OV-3 Operational Information Exchange Matrix. From the perspective of using C4ISR AF to model operations, the relationship between activities and the operational nodes that perform them is important, and needs to be captured regardless of the information flow. The OV-X Operational Node to Activity Relationship Diagram, discussed previously, captures these relationships.

Framework generally supported the development of the C4ISR AF products well. Its flexibility meant the development team could enhance Framework to capture the changes made to the C4ISR AF. However, Framework's focus is on developing the architecture products, and it provides little support for the management and preparation of data used to develop the products.

5 ACKNOWLEDGEMENTS

The authors wish to thank Moira Chin for her valuable guidance, support and advice in writing this paper.

6 BIBLIOGRAPHY

AWG. (1997). C4ISR Architecture Framework Version 2.0, US Department of Defence.

DoD. (1995). Joint Mission Essential Task List (JMETL) Development Handbook, US Department of Defence.

Kingston, G., Prekop, P., Chin, M., Jones, R., Kilpatrick, D. and Collier, P. (2000). Applying PTech Framework to Modelling Operational Architectures. Defence Operational Analysis Symposium 2000 (DSTO-GD-0239), Melbourne, Australia, DSTO.

Levis, A. H. and Wagenhals, L. W. (2000). C4ISR Architecture Design Process. In C4ISR Architecture Framework and Implementation Course, Canberra Australia.

MoD. (1999). Joint Essential Task List (JETL) Handbook, UK Ministry of Defence**:** PJHQ/7/7800.

Prekop, P., Kingston, G., Chin, M. and McCarty, A. (2001). A Review of Architecture Tools for the Australian Defence Force. Salisbury, South Australia, Defence Science and Technology Organisation**:** DSTO-TR-1139.