Australian Government
Department of Defence
Defence Science and
Technology Organisation

Future Proofing Computational Red Teaming

Scott Wheeler

Joint Operations Division
Defence Science and Technology Organisation

DSTO-TN-1109

ABSTRACT (U)

Computational Red Teaming describes the application of new and innovative analytic techniques, tools and methodologies in support of Red Teaming Activities. This report documents the formal design of such a tool, to inform the future direction of the DSTO research program. Requirements for an executable prototype are presented to construct the initial design, developed through the application of Systems Engineering practices. Recommendations to further develop the prototype are provided, with illustrative examples. When implemented, the prototype will deliver a Computational Red Teaming capability to assess technologies and systems, concepts and force structures; with the potential to inform better decisions in future Australian capability development.

Published by

Joint Operations Division
DSTO Defence Science and Technology Organisation
Fairbairn Business Park Department of Defence
Canberra ACT 2600 Australia

Telephone: (02) 6265 9111
Fax: (02) 6265 2741

AR-015-365
August 2012

Conditions of Release and Disposal

Future Proofing Computational Red Teaming

Executive Summary

Joint Operations Division established an initiative in Computational Red Teaming in 2010, under the Defence Science and Technology Organisation's Corporate Enabling Research Program (CERP). The program conducts fundamental research in the field, evaluates new and innovative analytic techniques, develops computational support tools and applies the methodologies of Red Teaming in support of activities and experimental campaigns.

This study follows from, and implements, the recommendations of two earlier scoping studies, delivered under the Computational Red Teaming initiative by Gowlett (2011) and Wheeler (2012). They recommend the development of a tool for the purposes of assessing Australian capability, concepts and force structures. Additional guidance for the CERP identifies the preferred domain of application for the tool should be in support of study and analysis for countering Improvised Explosive Devices.

Our report documents the start of the formal design process for a Computational Red Teaming Tool, in line with the existing program of work, and to inform the future direction of the corporate research program. Conceptual requirements for an executable prototype are presented and a design developed through the application of Systems Engineering practices. Illustrative examples for possible implementations of the design are also provided together with an example case study for application to the CIED domain.

The proposed prototype is to demonstrate the proof-of-concept. Further design work is required. However, should the prototype be successful, a wider program of research could be initiated. If implemented, the prototype will deliver a Computational Red Teaming capability to assess technologies and systems, concepts and force structures; with the potential to inform better decisions in future Australian capability development.

The following recommendation is advised, for forward work planning within the Joint Operations Division's Computational Red Teaming Task.

Recommendation 1.

The Computational Red Teaming Program continues to develop the prototype design presented in this report. Progression of this design to the next stage requires:

- selection of a case study or problem to be addressed in the CIED domain;
- identification of stakeholders, clients and end-users;
- development of use-cases, outlining how those end-users are to interact with the software; and
- trials of those interactions through example Graphical User Interfaces (GUIs).

Should this recommendation be supported, a team of at least four major skill sets will be required. These include: software design; systems engineering; human-factors; and program management. The team would take the initial conceptual design presented in this document and develop the customer requirements specification. This requirements specification will guide the development of this concept into the prototype.[1]

Finally, the purpose of this report is to deliver a design for a Computational Red Teaming software tool. To this end, the science of adversarial reasoning component has been less important than the strength of the design. We then present one further recommendation.

Recommendation 2.

The Computational Red Teaming Program initiate a study to proceed this work. The study will report on follow-on options to implement the adversarial reasoning and logic components of the design. In doing so, the relevant techniques from the discipline of Machine Learning and Artificial Intelligence will be considered. This study will be conducted in parallel with Recommendation 1 and its outcomes will inform the implementation of the design.

References:

P Gowlett (2011). *Moving Forward with Computational Red Teaming*. DSTO General Document, DSTO-GD-0630. Joint Operations Division, Fairbairn, Canberra

S Wheeler (2012). *Moving Forward with Computational Red Teaming*. DSTO Technical Report, DSTO-TN-1104. Joint Operations Division, Fairbairn, Canberra.

[1] It will also be the task of this team to develop a Test and Evaluation plan. This plan will include verification and validation of the software and introduce requirements for functional performance testing. This should include requirements for analytical products; such as operational measures of performance, application of statistical techniques, and graphical and visual outputs.

Author

Scott Wheeler
Joint Operations Division

Scott Wheeler joined DSTO as a Research Scientist after completing a PhD in mathematics at the University of Adelaide. Scott previously managed the Complex Adaptive Systems Task in Land Operations Division at the DSTO Edinburgh site before moving to the Defence Systems Analysis Division in Canberra's Russell Offices. In Russell, Scott worked in the domain of Capability Analysis and was the Science Advisor to the Missile Defence Coordination Office. More recently he represented Australia as the National Lead for TTCP AG-14 Complex Adaptive Systems. Scott was previously a Visiting Research Fellow at the University of NSW, Australian Defence Force Academy and now works for the Joint Operations Division DSTO in Fairbairn, Canberra.

This page is intentionally blank

Contents

ACRONYMS

1. INTRODUCTION **1**
1.1 Computational Red Teaming **1**
1.2 Scope **2**
1.3 Outline **2**

2. CONCEPTUAL DESIGN **4**
2.1 A Systems Engineering Approach **4**
2.2 Design Brief **4**
2.3 Statement of Operational Need **5**
2.3.1 Models for Capability Assessment 5
2.3.2 Applications of Capability Assessment 7
2.3.3 Design Implications 8
2.4 Functional Analysis **8**
2.5 Component Allocation **9**
2.6 Product Synthesis **11**

3. VISUAL EXAMPLES **12**
3.1 Top-Down Design Methodology and a Bottom-Up View **12**
3.2 Implementing the Entity Definition Framework **12**
3.3 Execution of Model **18**

4. FUTURE PROOFING CRT **20**
4.1 Application of the Model to CIED domain **20**
4.2 Unclassified Case Study: Operation Slipper 2010 ..... **20**
4.3 Application of Computational Tools **22**
4.3.1 Agent Representations 23
4.3.2 Process Models and Architectures 23
4.3.3 Context and Scenarios 26
4.3.4 Adversarial Learning 26

5. RECOMMENDATIONS **28**
5.1 Recommendations **28**

6. ACKNOWLEDGEMENTS **30**

7. REFERENCES **31**

APPENDIX A: DESIGN BRIEF **34**

APPENDIX B: OPERATIONAL NEEDS **35**

APPENDIX C: FUNCTIONAL ANALYSIS 40

APPENDIX D: COMPONENT ALLOCATION 53

APPENDIX E: SUBJECT MATTER CONTRIBUTION 61

List of Tables

Table 1: Components of a scenario 6
Table 2: Summary of operational requirements 8
Table 3: Mentoring and Reconstruction Task Force (MRTF) 21
Table 4: Four key steps 23

List of Figures

Figure 1: ORBAT browser GUI 13
Figure 2: ORBAT Browser + System Browser = Entity 14
Figure 3: Systems browser GUI 15
Figure 4: Entity + Vignette Browser = Entity State 16
Figure 5: jSWAT interface 17
Figure 6: Entity State + Behaviours = Defines Model 18
Figure 7: Methodology to develop process models from doctrine (ibid Aerospace Concepts (2010)) 25
Figure 8: Legend - Enhanced functional flow block diagram 40
Figure 9: Top level design 41
Figure 10: Serial 1: Maintain database 44
Figure 11: Serial 1.1: Initialise database 46
Figure 12: Serial 1.2 Interrogate database 47
Figure 13: Serial 3: Configure model 48
Figure 14: Serials 3.1.1, 3.1.2, 3.1.3: Define entity; define environment; instantiate model 50
Figure 15: Serial 4: Execute model 51
Figure 16: Legend: Functional component allocation model 53
Figure 17: Component allocation for Serial 1.1: Initialise Database 54
Figure 18: Component allocation for Serial 1.2: Initialise Database 54
Figure 19: Component allocation for Serial 2: Render Visuals 56
Figure 20: Component allocation for Serial 3.1: Define Entity 57
Figure 21: Component allocation for Serial 3.2: Define Environment 58
Figure 22: Component allocation for Serial 3.3: Instantiate Model 59
Figure 23: Component allocation for Serial 4.1: Update State 60

Acronyms

Acronym	Definition
ADF	Australian Defence Force
CERP	Corporate Enabling Research Program
CIED	Counter Improvised Explosive Device
CRT	Computational Red Teaming
DSTO	Defence Science and Technology Organisation
GUI	Graphical User Interface
IED	Improvised Explosive Device
JDSC	Joint Decision Support Centre
JOD	Joint Operations Division
JTF	Joint Task Force
jSWAT[1]	Joint Seminar Wargame Adjudication Tool
MRTF	Mentoring and Reconstruction Task Force
OPFOR	Opposing Force
ORBAT	Order of Battle
TTP	Tactic, Technique and Procedure

[1] Ibid (name of software adopts lower case 'j').

This page is intentionally blank

1. Introduction

1.1 Computational Red Teaming

The term *Computational Red Teaming* (CRT) has recently arisen within the literature to describe the application of new and innovative analytic techniques, tools and methodologies in support of Red Teaming Activities. This approach introduces a novel element to Red Teaming which is yet to be exploited; proposing to reduce risk and increase opportunities through computation. Central to the field is an objective to enhance the quality of decision making by "increasing the degree of rigor that can be brought to bear on complex problems." (Gowlett, 2011).

The Joint Operations Division (JOD), of the Defence Science and Technology Organisation (DSTO), established an initiative to future-proof defence against threats posed by Improvised Explosive Devices (IED) and those who employ them. It is under this framework that the CRT initiative was developed, and subsequently consolidated under its Corporate Enabling Research Program (CERP).

While the application area has largely stayed the same (with some extension into informing Australian capability development projects; Wheeler, 2010a, 2010b) the program has since matured. Research expertise in Red Teaming has been brought together under one banner for the first time in JOD. This initiative has consolidated the CERP and fostered a community of practice in the domain.

Subsequently, the CERP has expanded and now includes scope for fundamental research in the field (Antwerpen & Bowley, 2011; Wheeler, 2012), the evaluation of new and innovative analytic techniques (Rajesh et al, 2010; Wheeler, 2011), development of computational support tools in-house (Hossain et al, 2011) and the application of methodologies from Red Teaming in support of activities and experimental campaigns (Sayers & McKay, 2011).

However, there remains some debate as to the appropriate interplay between human-based exercises and computational methods of support (Skroch, 2009). Within the literature, the specific nature of support and the roles or functions computational methods can fulfil are also under discussion.

Notwithstanding, the CRT initiative investigates opportunities to complement human-based Red Teaming exercises (as independent investigation), to augment those exercises (in situ), and to replace humans in Red Teaming exercises (in part or whole). Wheeler (2012) defines CRT as follows.

> *Computational Red Teaming is the science concerned with the provision of analytic tools, in support to Red Teaming, for the purposes of improving the outcome of its application.*

The CRT initiative supports all potential applications for the discipline. This perspective has largely followed from the inaugural work in the CERP by Gowlett's 2011 report *Moving Forward with Computational Red Teaming* and the subsequent review by Wheeler (2012). Gowlett developed initial recommendations for the program, its scope and direction. Wheeler conducted a detailed study of CRT as a working concept and outlined a need for practical set of tools which could be rapidly applied in specific targeted studies across the scope of application for computational techniques.

1.2 Scope

These recommendations align with guidance in the 2011-2012 financial year for the CERP. The guidance identified a need for a CRT model akin to a computational toolkit supporting Red Teaming. It is suggested that this model should be an executable prototype; designed for the assessment of ADF capability. The preferred application domain for the program is to study ADF capacity for countering IEDs.

This report documents the initial conceptual design for a CRT tool, inline with the existing program of work, and to inform the future direction of the research program. Requirements for an executable prototype are presented and a design developed through the application of Systems Engineering practices. Illustrative examples for possible implementations of the design are also provided.

The proposed prototype is to demonstrate the proof-of-concept. Further development of the design is required. However, should the prototype be successful, a wider program of research could be initiated. If implemented, the prototype will deliver a CRT capability to assess technologies and systems, concepts and force structures; with the potential to inform better decisions in future Australian capability development.

1.3 Outline

This report is structured in four sections:

a. initial conceptual design for a CRT tool;
b. illustrative examples;
c. case study;
d. delivery of recommendations to the JOD program.

Section 2 outlines the initial conceptual design, from first principles, for a generic CRT tool. It begins by discussing the requirements to develop a system to assess Australian capability in countering IEDs. This discusses the principles behind models for analysing operating concepts, technology and systems, and structures. The proposed design for the system is then presented, together with textual descriptions of each component and function.

Having established the base design, Section 3 provides visual examples of possible implementations. The examples are presented as Graphical User Interfaces and this section explains how those interfaces work together to implement the design of Section 2.

Interested readers may wish to skip directly to this section to review a potential final product before visiting the underlying design.[2]

Section 4 presents an unclassified case study, in which an application of the prototype tools is posed, to study the capability employed in Operation Slipper. This case study is useful because it shows how the initial conceptual design can be employed in the Counter IED (CIED) domain. This will demonstrate the potential utility of the design concept.

Recommendations are presented in Section 5.

[2] The report is presented in order of design. In a top-down process (such as the one adopted in this report) virtual functions of the design are developed before their implementation with physical components. However, humans often prefer a bottom-up approach where the design can be explained explicitly in terms of the product (as opposed to the capacity to perform functions). Pictures of a potential working product will then aid in comprehension and convey context. Thus, the reader may wish to skip through Section 3 prior to Section 2.

2. Conceptual Design

2.1 A Systems Engineering Approach

In the following sections, a Systems Engineering approach is adopted to design a concept for the prototype software toolset. It is not the intention of this report to explain the science of Systems Engineering. However, for clarity, a brief explanation of the approach is included. The Systems Engineering methodology, applied within this report, is comprised of five stages:

1. Design Brief.
2. Statement of Operational Needs.
3. Functional Analysis.
4. Component Allocation.
5. Product Synthesis.

2.2 Design Brief

The Design Brief is the first step of the conceptual systems design process. This brief describes the mission for the system in plain terms. In short, it encapsulates the customer's vision for the system and its development. For the purposes of this report, the CRT initiative has outlined three key concepts. First, that a toolset be developed for CRT. Second, that the application of this toolset be aimed at the CIED domain. Third, that the toolset be capable of assessing the capability of threats posed by adversaries and the capability available to the ADF to counter those threats.

The corresponding Design Brief, as a Systems Engineering Product, is enclosed in Appendix A. The following root clauses are extracted from the Design Brief and are presented in summary here:

- JOD leads a CRT Program.
- The Program is run in support to the ADF.
- The Program objective is to counter IEDs and insurgency.
- CRT will assist The Program achieve its objective.
- The CRT capability will assess technologies and systems for CIED operations.
- The CRT capability will assess concepts and TTPs for CIED operations.
- The CRT capability will assess force structures for CIED operations.
- A Suite of software tools will be developed for this purpose.
- The Suite will be developed through conceptual design.

- The Suite will be developed as a prototype for evaluation.
- The Suite will be employed to inform improved decision making in support to Australian military operations and capability development.

2.3 Statement of Operational Need

The statement of Operational Needs identifies customer needs and requirements, and follows directly from the Design Brief. Unlike the Design Brief, it is a semi-structured product and is usually written as a series of formal clauses. These clauses may describe operational distribution or deployment, usage profiles, performance parameters, utilisation requirements, effectiveness and cost, operational life, environmental considerations, and others.

However, the statement of operational need is not a systems specification. Operational Needs are independent of solution. This means that the operational need describes a capability or function that the system must perform for the customer but does not stipulate how that capability or function is to be implemented. To develop this product, we first need to review the modelling options and applications, in the CIED domain, for capability development. This provides additional guidance to the Design Brief and ensures the virtual design is practical, implementable, and fit-for-purpose.

2.3.1 Models for Capability Assessment

There are many different (and equally valid) approaches to developing a CRT model for capability assessment. These depend somewhat on individual preference but more generally it is also true that the broad requirements in implementing different types of models vary considerably.[3] However, in all instances, three inputs are always required of the model:

1. Specifications of technologies and capabilities within scope of the model, their attributes, properties, and functions and performance characteristics.
2. Description of the environment and context.
3. Statement of the behaviours of all objects, responses to stimuli and environment, and concepts of employment and operation for capabilities.

In the first instance, specification of physical characteristics associated with known technologies and capabilities is a comparatively straightforward task. Technical details of platforms, sensors, weapons and other systems can be defined in quantifiable terms and translated into the model, as appropriate, for the task at hand. Where the systems to be tested are still in development, prototype, or otherwise untested, then performance characteristics might instead be estimated.

[3] It is not the intention of this report to complete a formal systems engineering process for selection of the specific approaches for each application class. Instead, we will broadly outline some of the associated issues and develop a plan of approach.

When the capability to be modelled is a living organism, the corresponding set of features to be specified is somewhat more variable. Irrespective, there is a vast body of research to draw from when building a model to incorporate these features. For example, there are disciplines devoted to the study of biometrics, human performance systems, and the like. Within DSTO, the Human Protection and Performance Division contains the expertise to support the modelling of living capabilities.

Modelling of an environment and wider strategic context is a slightly more complex task. Wheeler (2010b) identifies the key components of a well developed scenario, for military applications. These components are provided in Table 1 below.

Table 1: Components of a scenario

Category	**Product**
Context	Strategic Narrative
	Government Guidance
Environment	Operational Environment
Participants	Strategic Intent (other forces)
	Government Objectives
	Threat Assessment
	ORBATs
	Force Dispositions
	Command Arrangements
	CONOPS
	ROE
Sequencing	Storyboard
	Phasings
	Vignettes

The behaviour of each entity in the system must be appropriately defined for input to the model. Entities and objects have internalised behaviours, which might be carried out in the absence of opposition. Capabilities for example have concepts of employment and operation. Entities also possess tailored responses. These might be carried out as a result of pressure, from external stimuli and the environment.

In contrast to the physical properties of a system; the behaviours of a system, entity, object or organism are difficult to describe. The specific implementation of behaviours can be ambiguous and open to subjective interpretation. However, not all applications directly require, or otherwise depend on, complex and subjective behavioural rules. Applications of Red Teaming models include the spread of disease and contagion, study of natural disasters including wildfires and floods, contamination effects of CBRN reagents, emergency response situational studies, and riot control and crowd dynamics. In such situations, the aggregate behaviours of the participating entities are well understood.

2.3.2 Applications of Capability Assessment

The previous section has outlined some of the difficulties in implementing a Red Teaming model for capability assessment. Given the objective, being to develop a prototype, it is preferable to develop a simple objective model in the first instance. This simple model would have the smallest set of subjective parameters with the aim to test the properties of the system of interest and not the behaviours.

To elaborate on the distinction between the range of possibilities in the Design Brief; a model for capability assessment could be used to conceptualise about:

1) how capability can be used;
2) which capability should then be purchased; and
3) the force structures to support it.

These three concepts can also be expressed more specifically in terms of targeted studies. A model for capability assessment could be used directly to study three things:

- *Operating concepts and processes*. Outcomes of such studies might direct the development of future joint and service concepts, doctrine and Tactics, Techniques and Procedures, and support training programs.
- *The impact of technology or capability in both current and future operations*. Outcomes of such studies might inform defence investment and acquisition, contribute to an intelligence picture when applied to adversaries, and support military plans when applied to support operations in theatre.
- *Alternative force structures*. Outcomes of such studies might identify breaking points and capability gaps in the force; and inform capability development in the transition and evolution of the force 'in-being' to alternative structures and aspirational states.

The study of operating concepts and processes focuses on the set of behaviours of a system. As explained in the previous section, this can become almost arbitrarily complex; being difficult to describe, ambiguous and open to subjective interpretation.

In contrast, the study of technology or capability impact is largely objective. Such a model focuses on the set of properties of the system. Again, as outlined above, technical specification of platforms, sensors, weapons and other systems can be defined in quantifiable terms. It then is a simpler task to implement a technology impact study than a study of operating concepts and processes.

The study of alternative force structures is of a level of difficulty above either of the two other applications because it implies both operating concepts (for many systems) and technologies are implemented. It also assumes that the fundamental relationships between the systems and force structures are captured.

2.3.3 Design Implications

Given the Design Brief of Section 2.2 and the discussion of options in application it is now possible to develop a set of Operational Needs. This set is not unique and will change over time with customer interaction. However, the conceptual design process must start somewhere and the initial Operational Needs are provided in Appendix B.

Key summary points from Appendix B are provided in the table below.

Table 2: Summary of operational requirements

Operational Issue	Prototype Software	Final Product
Access	Local	Remote via Internet Browser Access on Intranet
Availability	Stable build within 2 weeks	75 hours a fortnight
Clients	CIED TF, CDG, VCDF	
Instantiation	JOD Analysts build database and initialise	Information repositories fully maintained by client
Operated By	DSTO Analysts	Customer
Performs Functions	Assess technology Assess concepts Assess force structures	
Support	JOD Analysts	Helpdesk hotline Training courses
Standard Operating Environment	JOD Server Portable Device (Laptop)	DRN DSN
Visualisation	Overhead Projector Custom Docking	Multiple Monitor Custom Docking

2.4 Functional Analysis

Functional analysis is the process of translating system requirements into detailed design criteria. The result is a specification of the system architecture. This architecture describes the system in terms of specific actions that are necessary to achieve a given objective. The method by which those actions are performed is not stated. The analysis breaks down the Operational Needs into lower level requirements including functions of design, production, distribution, operation, maintenance and disposal.

For our purposes, the design functions concentrate on the operation of the Prototype. The 'what' the software should do is converted into 'how' the Prototype should do it. The

design is never unique. It is up to the design engineer to specify an initial conceptual design, which will be discussed with the client. This report provides such a conceptual design, which has not yet undergone client review. Although there are many standards for architectural design, at this stage is most practical to adopt the simplest. Thus, this report employs an Enhanced Functional Flow Block Model (sometimes referred to as an activity model or OV-5 (US DoD, 2007a)) to represent the architecture. The functional analysis is provided in Appendix C.

The design in Appendix C is necessarily lengthy. A summary of the conceptual design is then provided below for interested readers. The design is comprised of four key functions, with a component breakdown as shown below:

1. Database management.
    - Initialising.
    - Interrogating.
2. Rendering visuals.
3. Configuring models.
    - Entities.
    - Environments.
    - Models.
4. Executing simulations.
    - State changes.

The key points of note are:

- The database management function plays a pivotal role in the design. The implementation will be modular, as detailed in the Operational Needs. Hence, each independent component of the software will interface and synchronise through a central database.
- Visualisations are essential to customer use. The graphical user interface will facilitate the use of the software and convey information to clients.
- There are multiple stages in configuring a model. The properties and behaviours of the entities in the model must be defined, and the modelling environment including metadata needs to be specified, prior to simulation.
- Simulation of the model is defined as a functional change in the state of the model. This design facilitates logging of state, replay, and post-analysis.

2.5 Component Allocation

Trade-off studies are often commissioned to determine the specific balance of hardware, software, facilities and staff resources needed for each function in the Functional Analysis. The purpose of these studies is to identify constraints and options, in resourcing; when allocating components to implement the functions in the conceptual design.

The component allocation for our conceptual design is presented in Appendix D. We present an overview of the key findings of the allocation below.

There are three types of components in our design. These are:

1. People and teams.
    - Database design and administrator.
    - Analyst.
    - Model developer.
2. Agents.
    - Database.
    - Visuals.
3. Frameworks.
    - Model, instance and state.
    - Layout and user-interface definition.
    - Entity definition.
    - Environment definition.
    - Metadata definition.

The first category, people and teams, represents real staff that cannot otherwise be replaced by automated processes. A database must be established, and it will need to be well designed and administered. Analysts will employ the software during live activities and those models used in the activity must also be constructed.

The second category, agents, are automated processes which represent services which must be provided. In this case, a database service is likely to be provided simply by installing freely available software such as MySQL or proprietary software such as Microsoft Access. Visuals must be also be rendered. The agent here refers to the Advance Programming Interface adopted by the software and its corresponding graphical and visual functions. It is likely that the development language will be either Java or Microsoft C#. Both provide integrated graphical design software and advanced programming interfaces.

The final category, frameworks, describe the interfaces themselves. This implies that someone (outside the people and teams) has defined and implemented the graphical interfaces. Detailed studies are usually conducted to determine the form of these frameworks. These include use-case studies, live and offline tests, and a host of client oriented verification and validation studies.

In summary, we can make a broad categorisation of the types of skills required to proceed with the design of the prototype. Some of these follow directly from the component allocation, others are implicit to the design. These include:

- Analysts – to implement models, run simulations, and analyse results.
- Client Liaison Officer – to engage the clients.

- Database Administrators – to implement, populate and support the database.
- Human Factors Scientists – to conduct usability studies and design the interfaces.
- Project Managers – to ensure delivery of product.
- Software Developers – to deliver the software.
- Systems Engineers – to continue the conceptual design.

2.6 Product Synthesis

Product synthesis is the act of developing a number of candidate options for the conceptual design and comparing the implementation benefits, costs and risks. As with the Component Allocation, trade-off studies are often employed. The means by which candidate options are compared is often complex, involving multi-criteria optimisation. In addition, the factors for consideration and the subsequent metrics for assessment of those factors are difficult to derive.

Our design remains at the conceptual level. Product synthesis is somewhat out of scope. To continue the design process, a number of steps are required. First, a case study or problem to be addressed in the CIED domain must be identified. This also relates to the identification of stakeholders, clients and end-users. Given those clients, use-cases can be developed to outline how those end-users are to interact with the software. Finally, trials of those interactions can be conducted.

This proposal will include a Test and Evaluation plan. The plan will include verification and validation of the software and introduce requirements for functional performance testing. This should include requirements for analytical products; such as operational measures of performance, application of statistical techniques, and graphical and visual outputs.

If these products are developed, in addition to the trade-off studies, then product synthesis can be attempted. In the meantime, it is still possible to describe one potential solution which would meet the needs of the CRT CERP. This is not considered part of the formal design process because the prerequisite stages, detailed above, have not yet been completed. However, it is valuable because it presents a concrete and tangible example of how the system might work. This synthesis is presented in Section 3. A subsequent case study using this synthesis is presented in Section 4.

3. Visual Examples

3.1 Top-Down Design Methodology and a Bottom-Up View

Section 2 describes the formal model for the prototype design. This description includes the virtual functions and their instantiation in the form of components. However, the component instantiations are still necessarily abstract. This prevents the design from becoming over specified and constraining. The model design is flexible and will accommodate a range of possible specific physical implementations.

However, it is also useful to explain the model in more detail. This explanation would not be a formal part of the design process but would aid in comprehension. In a way, it provides an alternative perspective on the model; that being, from the bottom up. Detailed visual cues can help a systems designer understand the intent behind the model. This intent can help the design by exploring the way users would like to interact with the model and is important for usability considerations.

This section applies a Rapid Prototyping 'like' approach to present examples of a prototype implementation for some of the components from Section 2. We present potential Graphical User Interfaces (GUIs) for the component and demonstrate some of the products described earlier through diagrammatic and visual means.

3.2 Implementing the Entity Definition Framework

For the purposes of demonstration, this section explores the concept for the Entity Definition Framework. This framework was presented in Section 2. It provides the interface by which entity properties and behaviours are defined.

Consider for the moment the idea of visually interacting with military units (entities). The end-user would normally expect this interaction to be facilitated through a GUI. The interface should be functional and practical, for use by its end-users (the military) and leverage from their background, expectations and experience. It is easy to conclude that a 'good' interface would be displayed in the symbology that defence personnel are familiar with. Standards such as *MIL-STD-2525 (C)* (US DoD, 2007b) and *NATO AAP (C) / STANAG 2019 (6)* (NATO, 2011) exist for this purpose.

Further, defence forces around the world have already adopted a template for symbolically visualising military units. This template is called the Order of Battle (ORBAT). At its simplest, an ORBAT is a list of military units together with the number of each available. Detailed ORBATS include additional information including equipment and loading, and other special and reserved tags. The visualisation of the ORBAT can be textual but is more often displayed as an organisational chart.

Figure 1 displays an indicative graphical user interface for the first part of the entity definition framework. An indicative ORBAT is presented in the chart in the middle of the figure.

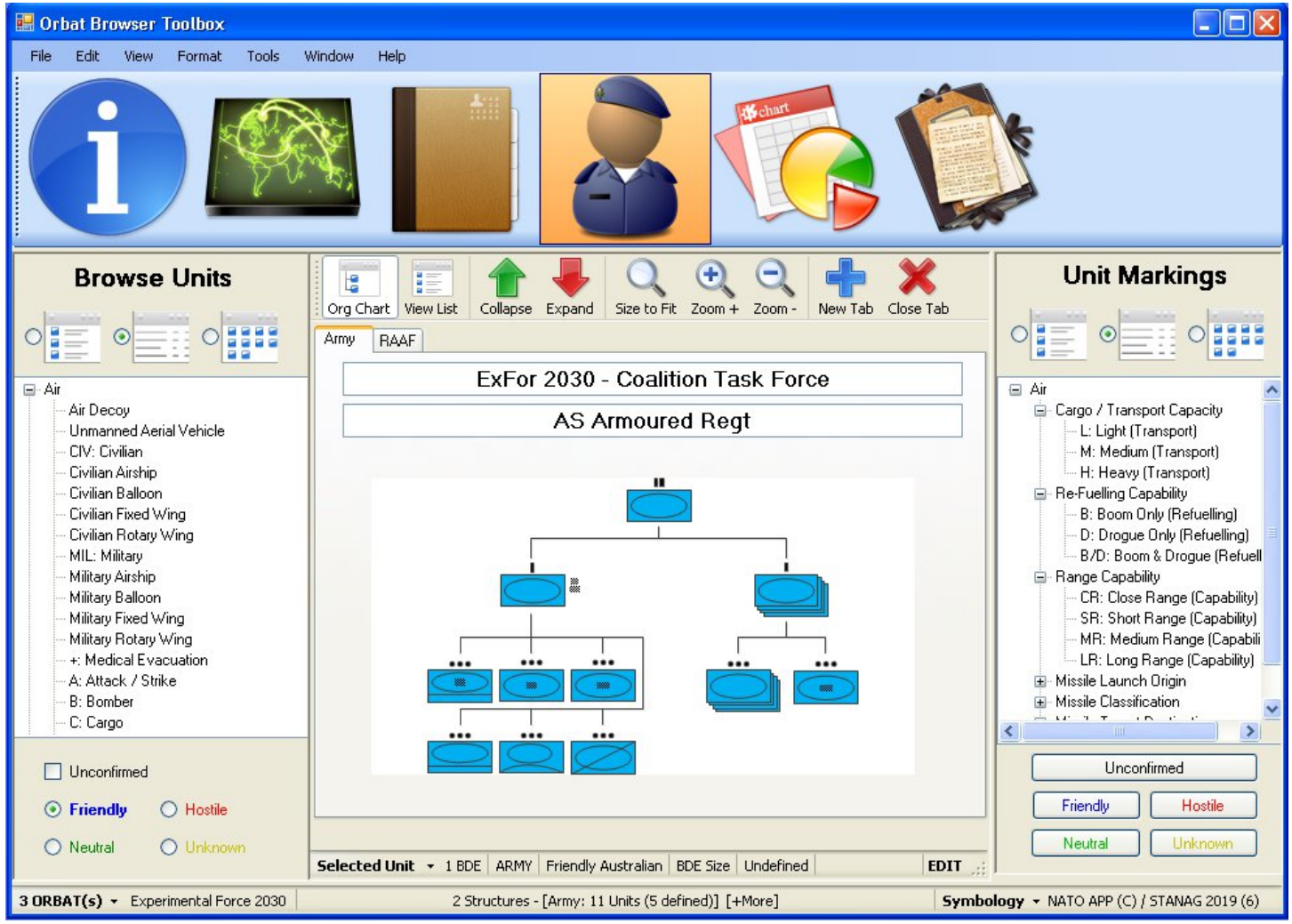

Figure 1: ORBAT browser GUI

The example provided above demonstrates a modular design concept. This basic *ORBAT Browser* can be developed as a product, which is useful in a variety of experimental activities in its own right. The brower is integrated with the other design components through the database implementation, running the back-end.

The ORBAT Brower facilitates the definition of a force structure. For the purposes of implementing component F.6 'Entity Definition Framework' (Refer Appendix D), this is itself insufficient. However, from this point it is easy to grasp the next step in the implementation. An additional user interface, a *Systems Browser*, can be implemented. The attributes of each of the units in the ORBAT then inherit a set of default properties (stored in a library repository within a database) and can also be accessed, viewed and customised in the Systems Browser. Figure 3 illustrates a potential interface for the systems browser.

Once more, the Systems Browser is a product which can be used in a variety of different experimental activities and is again a modular component. The strength of the combination of the two modular components is that they can be developed semi-independently. Integration of the two products is ensured through the sharing the database in the core design.

It is crucial to note that implementing the functionality of the ORBAT Browser and Systems Browser cannot be avoided.[4] The design presented in Section 2 does not necessarily need to implement these two specific products. However, the Entity Definition Framework must be implemented. All of the functions supported in the ORBAT Browser and Systems Browser will be then developed. If so, those functions may as well be developed into products that have utility in there own right.

The combination of the ORBAT Browser and Systems Browser, with a little additional work, then implement component F.6 Entity Definition Framework. The basic concept is illustrated in Figure 2.

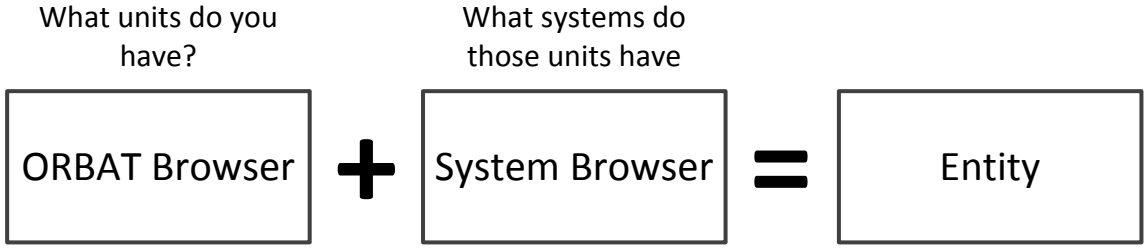

Figure 2: ORBAT Browser + System Browser = Entity

[4] In the short term, the prototype may simply read and write text files. The user could pre-define the structures and entities in these files and this is sufficient to run the prototype. However, in the long term, a text file solution is impractical and a graphical solution will be needed for deployment in a client space. Using best practice, the user interfaces can also support debugging and testing which would otherwise not be possible through a text file.

Figure 3: Systems browser GUI

Assume now that F.6 'Environment Definition Framework' was also implemented; lets say, by a *Vignette Browser*. The Vignette Browser can display geo-spatial map data and allow the user to drag and drop units onto the space. It would also permit users to enter information regarding tactical disposition, condition reports and other specific status information. Such a tool would again be useful as a stand-alone product in military seminar war-gaming and follow a modular design concept. Then, the combination of an entity and its environment define the state of the entity. This idea is illustrated in Figure 4 below.

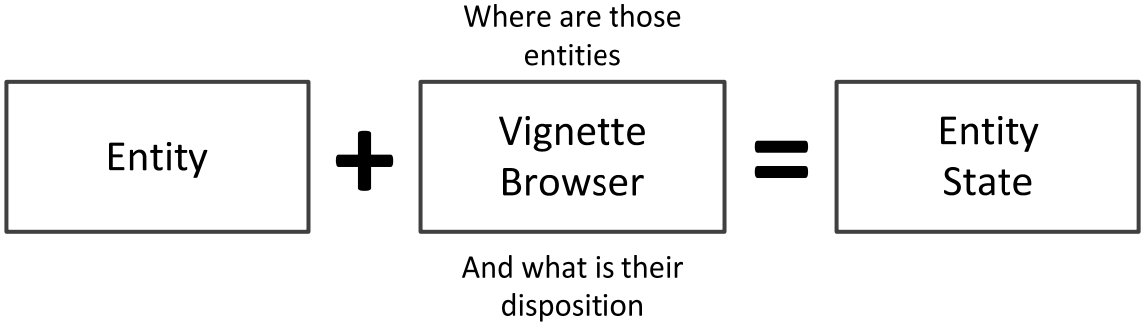

Figure 4: Entity + Vignette Browser = Entity State

There are many tools already in existence which provide the functionality of a Vignette Browser. The Joint Seminar Wargame Adjudication Tool (jSWAT)[5] provides an example. Figure 5 displays a screenshot of the tool; taken from Robinson (2011). This tool is already in existence and is used by the Australian Defence Force in its Army Experimental Framework (Australian Army, 2000).

[5] Ibid (name of software adopts a lower case 'j').

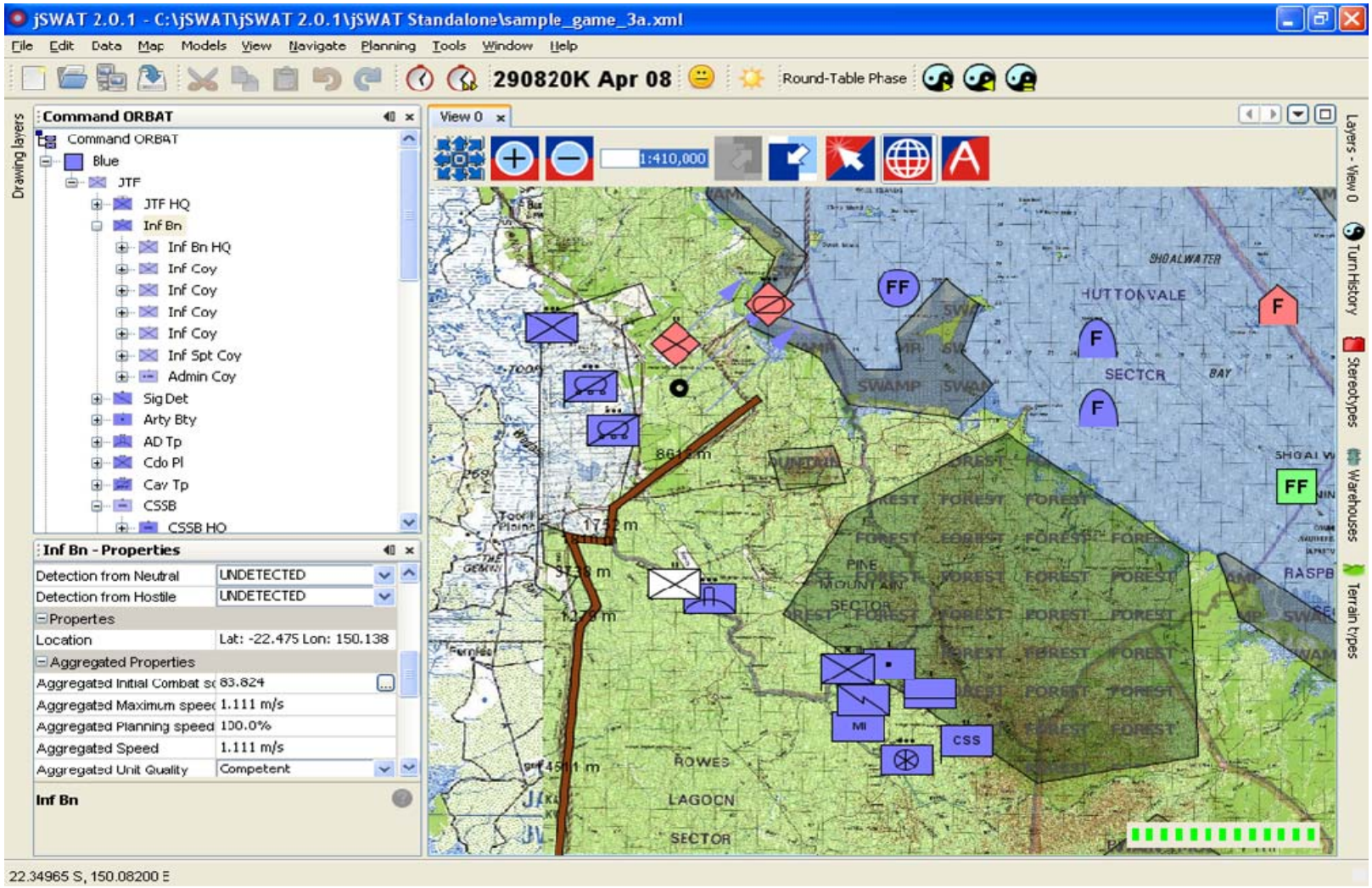

Figure 5: jSWAT interface

A final interface must be implemented to define the behaviours for each entity. These can be taken from Tactics, Techniques and Procedures (TTPs) and doctrine; modified by context, Rules of Engagement and Command Guidance. It would normally be possible to codify these into a set of simple rules. More complex rules can be expressed in terms of process flow diagrams / enhanced functional flow diagrams. This is presented in Figure 6 below.

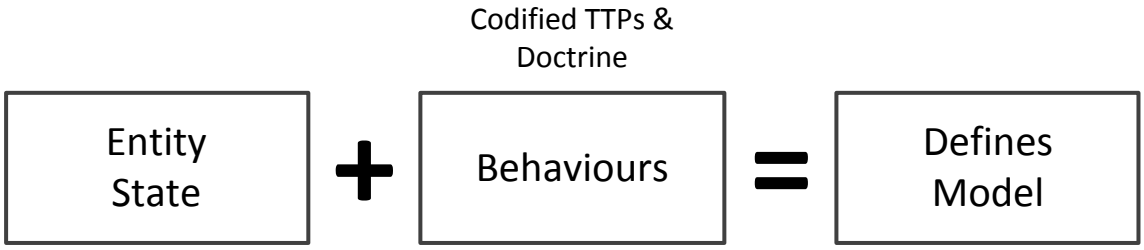

Figure 6: Entity State + Behaviours = Defines Model

The model is now ready for execution.

3.3 Execution of Model

At this point it behoves the interested reader to ask how precisely the model is to be executed. The execution methodology for the model is largely dependent on the implementation of the behaviours; which we have not specifically addressed in the previous section.

At its simplest, the behavioural model may be tacitly implemented in the subject matter knowledge and experience of the Red Team. This approach is common in military seminar wargames. In this case, the codification of knowledge about entities is restricted to the properties and attributes of the units or platforms. At fixed intervals and decisive points in the wargame, the participants (or supporting analysts) will move entities across the playing board according to their known experiences and knowledge of those entities capabilities.[6]

One alternative approach is to codify unit behaviours directly into the model itself. This step of the process is an application of Artificial Intelligence and provides a user with an executable model. One of the first successful Artificial Intelligence techniques for this purpose is known as *Expert Systems*.[7] A Expert System is "a computer system that emulates the decision-making ability of a human expert."[8] The concept of the Expert System is interesting because it preposes that the knowledge, experience and reasoning of a human can be captured as a set of rules, which are themselves codified into a

[6] The jSWAT tool in Figure 5 is an example of exactly such a tool.
[7] See for example Jackson (1998).
[8] Wikipedia, *Expert System*. http://en.wikipedia.org/wiki/Expert_system accessed 1 July 2012.

'knowledge base' within computer code.[9] Today, we accept the limitations of Artificial Intelligence, especially in regards to capturing human decision-making capabilities. Notwithstanding, this approach has proven to be practical (and successful) in many applications. Languages such as *Prolog*[10] excel in implementing this approach.

The next step in implementing an Expert System requires the model, or system, to learn which of the 'expert' rules to apply in given contexts.[11] Given such a set of codified rules, there exist a host of techniques from the field of *Machine Learning*[12] which can be applied.[13] These include.

- Artificial Neural Networks (Ghosh & Dehuri, 2004).
- Association Rule Learning (Hipp et al, 2000).
- Bayesian Networks (Guo & Hsu, 2002).
- Decision Tree Learning (Rokach & Maimon, 2005).
- Evolutionary Algorithms (Ghosh & Dehuri, 2004)
- Reinforcement Learning (Kaelbling et al, 1996).
- Support Vector Machines (Wang, 2008).

Each approach has its own associated strengths and weaknesses, as well as particular nuances of application. However, the research problem is far from insurmountable and each technique listed above is well established within the literature.

[9] Some problems do not require a library of rules to be pre-defined but only structured. The classic example of such a problem includes optimisation or search over a parameter space. In this instance, only a set of numerical values need be codified.
[10] See Wikipedia, *Prolog*. http://en.wikipedia.org/wiki/Prolog, accessed 1 July 2012.
[11] Within a military domain, the codification of behavioural rules is not as difficult as it may seem. These can be taken from TTPs and doctrine, Rules of Engagement and Command Guidance. The context in which these rules apply is generally also well understood.
[12] See for example Mitchell (1997).
[13] This list is indicative only. See also Wu et al (2008), and similar papers, for a review of leading algorithms in data mining, search and optimisation.

4. Future Proofing CRT

4.1 Application of the Model to CIED domain

Section 1 defined the purpose of this research; being to develop a system to assess Australian capability in countering IEDs. The design of the system in Section 2 is necessarily abstract, to accommodate a range of possible future CRT tools for this purpose. However, it remains to discuss how that design, together with the illustrative examples provided above, can be specifically employed to the CIED domain. This will demonstrate the utility of the design.

Section 2 provided three examples of potential applications for this suite of tools; namely, in the study of operating concepts and processes, the impact of technology or capability in both current and future operations, and impact of alternative force structures. To demonstrate the design, we must first create an example of a particular study from these three options.

Let us assume we wish to explore the issue of threat assessment, to inform acquisition of measures for force protection. These measures would be issued to Australian troops on deployment for use in operations. The objective of employing the tools proposed by our design is then to assess the operational impact that these new measures offer. We might also wish to understand how weaknesses in the proposed measures might be exploited by adversaries.

4.2 Unclassified Case Study: Operation Slipper 2010

We will only refer to unclassified sources of information in this exercise. Let us use a case study from Operation Slipper in Afghanistan over 2010. The Australian-led Joint Task Force was reported on 8 February 2010 as the following.

Table 3: Mentoring and Reconstruction Task Force (MRTF)[14]

Army Elements
HQ JTF 633 (A)
HQ 1 RAR (Motorised Infantry)
A COY 1 RAR (Combat Team – CBT TM)
B COY 1 RAR
C COY 1 RAR
SQN 2 CAV REGT (ASLAV)
Engineer Task Group (ENGR TASK GP) - 3 CER
2 x Weapons Intelligence Team
DET 20 RISTA (UAV)
Force Support Unit (FSU)
Force Communications Unit (FCU)
Rotary Wing Group (ROTARY WG GP) 2 x CH47D
Special Operations Task Group (SOTG)
RAAF elements
RAAF Control & Reporting Centre
2 x AP-3C
3 x C-130J Hercules
Coalition Heron UAV Detachment (CHUD)

In 2010, the MRTF was tasked to:

a. Conduct route overwatch operations in order to gather information on IED network activities to support planning for disruption / interdiction of adversary IED activities;

b. Conduct Intelligence Preparation of the Battlefield in order to identify enemy assets, routes, routines and patterns;

c. Conduct Cordon & Search operations in order to interdict IED network activities;

d. Conduct route clearance in order to ensure Joint Manoeuvre; and

e. Validate Tactics, Techniques and Procedures in order to enhance Force Protection.

[14] http://en.wikipedia.org/wiki/Operation_Slipper correct as at 8 Feb 2010.

In fact, during 2009, the Minister for Defence commissioned a review of the Force Protection measures for Australian forces in Afghanistan. As an outcome of the review, additional capability was recommended for the MRTF, to "include:

- improved protection and firepower for Protected Mobility Vehicles;
- new night-fighting equipment;
- improved body armour;
- a new weapons system for the Special Operations Task Group;
- additional military working dogs; and
- a suite of improved intelligence and reconnaissance capabilities." (Rayner, 2010).

The question is then posed; to what extent did the capability recommended in the ministerial Force Protection Review of 2009 assist the MRTF defined in Table 2 to perform the tasks a. through e. on the previous page. This is, of course, a purely created question, for the purposes of our discussion. However, it is easy to imagine that analysis of a similarly posed question was conducted leading into the review and presentation of its findings to the minister.

4.3 Application of Computational Tools

We now apply the design framework established in Section 3 to the problem at hand. For ease of explanation we have divided our problem into four steps. These steps can be interpreted as research streams in a DSTO task, each delivering their own products and outcomes, yet each working together as an integrated program of work to answer our research question.

The integrated program is comprised of four streams of research:

1. Agent Representations.
2. Process Models and Architectures.
3. Context and Scenarios.
4. Adversarial Learning.

Each of the four streams can be categorized in two ways.

1. By principle focus:
    i. Description and representations of force elements;
    ii. Capture of processes and behavioural modelling.

2. By research methodology:
    i. Modelling (definitions, closed form expressions and processes);
    ii. Agent-simulations (numeric computation).

This concept is illustrated in Table 4 below.

Table 4: Four key steps

	Definition	**Simulation**
Force Elements	**1.** Agent Representations	**3.** Context & Scenarios
Processes & Behaviours	**2.** Process Models & Architectures	**4.** Adversarial Learning

4.3.1 Agent Representations

It is often assumed that because a force structure can be named (e.g. JTF 633(A)) that there is one unique and one shared understanding of the corresponding capability, structure and technologies within that force. However, within any discussion, there are multiple perspectives on each force structure because each force structure contains a combination of legacy systems, defined but yet to be acquired platforms, and virtual capability denoted by project name.

The limits to any individual, even senior decision makers, to fully comprehend the force structure and to express (or discuss) that comprehension in a military exercise, activity or experiment is limited. However, this level of knowledge is essential because many activities, where this is required, have objectives which involve the assessment or evaluation of the force structure and its ability to generate effects desired by government (in our case, these are the specific roles a. through e. for the MRTF defined earlier).

Step one in solving the problem posed in our case study of JTF 633(A) is to define the composition of the MRTF with the objective to translate, define, represent and ultimately model the force. Table 2 identifies the unit structure, but does not otherwise provide any inherent capacity to model those units. A complete ORBAT, with unit hierarchy and decomposition into subordinate units is required. Specific platforms, technologies and equipment must also be specified.

The ORBAT Browser in Figure 1 will help to define the units for modelling in a structured format. It will also be capable of capturing the original structure of the task force, prior to the additional capabilities received as a result of the ministerial Force Protection Review, and the structure of the force with its enhanced capabilities inserted. The Entity Browser in Figure 3 will define the specific properties, attributes and technologies possessed by each unit in the ORBAT. An enemy force can also be represented.

4.3.2 Process Models and Architectures

Process models can be developed for each of the first three tasks assigned to the MRTF: route overwatch; intelligence preparation of the battlefield, and cordon and search. These

process models follow from procedures encoded in doctrine and TTPs. The tactical activities for each of these three tasks are available in Wheeler (2010a). Further, the Operational Activity Model which is itself a process model is available in Aerospace Concepts (2010). Unfortunately, the specific tactical activities and corresponding activity models are classified, so examples cannot be provided within this report. However, Figure 7 provides an overview of the methodology.

Once the process models are defined for each task, a standard representation of the models and the mechanism can be adopted for its storage. For simplicity, we also assume that the concepts of operation, doctrine, TTPs for the new capabilities introduced as a result of the ministerial Force Protection Review does not differ from standard practice. We could assume otherwise and develop an entire new suite of process models. However, this is largely immaterial to our discussion.

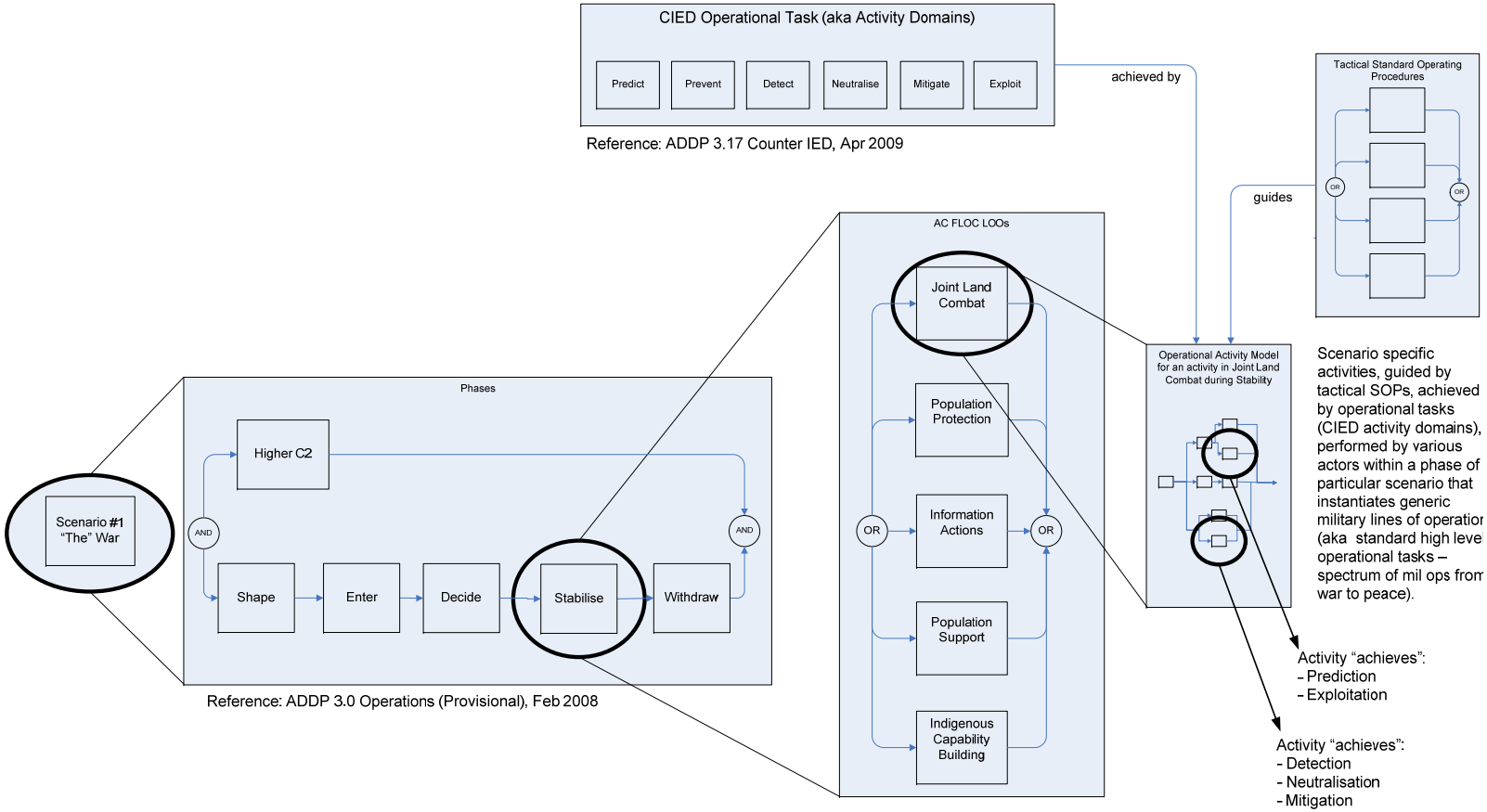

Figure 7: Methodology to develop process models from doctrine (ibid Aerospace Concepts (2010))

4.3.3 Context and Scenarios

JTF 633(A) is a term which describes the collection of units listed in Table 3. However, it also infers a situation; namely, the deployment to Afghanistan and the association with Operation Slipper. The concept being presented here is that military formations such as task forces exist within a context. To make sense of this, we often refer to this context in terms of a scenario. The scenario is itself defined according to Table 1 and a number of pre-defined scenarios for JTF 633(A) are presented in Wheeler (2010a). Unfortunately this is also classified and cannot be provided within this report.

Geospatial Information Systems are the most common means for the visualisation of military context. The Vignette Browser tool, illustrated in Figure 5 demonstrates the depth of information which can be expressed through visual means. Each unit in the ORBAT can be placed on the map and annotated with information describing its disposition, tactical readiness, or military state.

We have now defined the set of all units in the MRTF, the process models representing their behaviours, and situated those units within a scenario. At this stage, it is possible to automate a model using agent-based techniques. More importantly, the original research question can be addressed through simulation of the MRTF in the execution of its tasks. The performance of the original force, prior to the ministerial review, and the performance of the enhanced force, post the ministerial review, can be directly compared. Through this comparison, the operational impact of the new force protection measures can be objectively assessed. The paper by Wheeler (2006) provides an example of how different mixes of forces can be compared.

4.3.4 Adversarial Learning

The agent-based model proposed above introduces a credible defence context, in which agents are situated. Those agents are defined by unit structure from ORBATs and behaviours defined in terms of process models from doctrine and TTPs. However, there has not been any mention yet of a credible adversary.

In Section 4.1, we expressed the desire to understand how weaknesses in the measures proposed in the Minister's Force Protection Review might be exploited by adversaries. This particular inquiry can only be answered through an analysis of the interplay between forces, where the forces evolve to improve their operational performance. For example, the Minister's review includes improved protection for Protected Mobility Vehicles. Might insurgents be inclined to detonate larger IED charges in response? The review also includes upgrades to personal body armour. Might insurgents switch to armour piercing rounds or incendiary?

In executing the process models discussed in the previous two sub-sections, each agent has a number of operational and tactical choices to make. These may appear as branches in the process model and are entirely dependent on environment and situation. Likewise, any enemy force also has similar choices. Unlike the simplistic view expressed in the sub-section above, both the MRTF and the adversary force must be encoded such that their

agent representations learn from the operational and tactical choices made during execution of the model. Section 3.3 provided a list of techniques and methodologies which can be employed for this purpose.

5. Recommendations

5.1 Recommendations

This report documents the initial conceptual design for a CRT tool, inline with the existing program of work, and to inform the future direction of the corporate research program. Requirements for an executable prototype are presented and a design developed through the application of Systems Engineering practices. Illustrative examples for possible implementations of the design are also provided.

The proposed prototype is to demonstrate the proof-of-concept. Further design work is required. However, should the prototype be successful, a wider program of research could be initiated.[15] If implemented, the prototype will deliver a CRT capability to assess technologies and systems, concepts and force structures; with the potential to inform better decisions in future Australian capability development.

The following recommendation is advised, for forward work planning within the Joint Operations Division's CRT task.

Recommendation 1.

The CRT Program continues to develop the prototype design presented in this report. Progression of this design to the next stage requires:

- selection of a case study or problem to be addressed in the CIED domain;
- identification of stakeholders, clients and end-users;
- development of use-cases, outlining how those end-users are to interact with the software; and
- trials of those interactions through example GUIs.

Should this recommendation be supported, a team of at least four key skill sets will be required. These include: software design; systems engineering; human-factors; and program management. The team could take the initial conceptual design presented in this document and develop the customer requirements specification. This requirements specification will guide the development of this concept into the prototype.

It will also be the task of this team to develop a Test and Evaluation plan. This plan will include verification and validation of the software and introduce requirements for functional performance testing. This should include requirements for analytical products; such as operational measures of performance, application of statistical techniques, and graphical and visual outputs.

[15] The approach should be modular, such that a number of separate tools can be developed over time with minimal resourcing.

Finally, the purpose of this report is to deliver the first stages in the design for a CRT software tool. To this end, the science of adversarial reasoning component has been less important than the strength of the design. Section 3.3 has, however, provided an indicative roadmap for implementing this reasoning engine and has outlined some of the approaches that could be adopted.

A detailed review and comparative assessment of algorithms for Machine Learning and Artificial Intelligence is now called for. We then present one further recommendation.

Recommendation 2.

The CRT Program initiate a study to proceed this work. The study will report on follow-on options to implement the adversarial reasoning and logic components of the design. In doing so, the relevant techniques from the discipline of Machine Learning and Artificial Intelligence will be considered. This study will be conducted in parallel with Recommendation 1 and its outcomes will inform the implementation of the design.

6. Acknowledgements

Thanks are extended to all of those who contributed to this report. There have been almost too many to mention individually. However, special mention is provided to key personnel in Appendix E. Thanks are also extended specifically to my colleagues within the CRT task, both current and past.

7. References

Aerospace Concepts Pty Ltd (2010). *CIED Enhanced Functional Flow Block Model and Operational Activities for Middle East Area of Operations*. Classified contractors report to Defence Science and Technology Organisation, Canberra.

Australian Army (2000). *Army Experimental Framework*. Land Warfare Doctrine, LWD 8-2. Puckapunyal, Australia.

C van Antwerpen and D Bowley (2011). An Australian approach to Concept Development and Experimentation: Linking Strategy to Capability. *Journal of the Operations Research Society*, Vol 63, pp. 278-292.

A Ghosh and S Dehuri (2004). Evolutionary Algorithms for Multi-Criterion Optimization: A Survey. *International Journal of Computing & Information Sciences*. Vol 2(1), pp. 38-57.

P Gowlett (2011). *Moving Forward with Computational Red Teaming*. DSTO General Document, DSTO-GD-0630. Joint Operations Division, Fairbairn, Canberra

H Guo & W Hsu (2002). A Survey of Algorithms for Real-time Bayesian Network Inference. *Artificial Intelligence*. Vol 1, pp. 1-12.

J Hipp, U Güntzer and G Nakhaeizadeh (2000). Algorithms for Association Rule Mining: A General Survey and Comparison. *SIGKDD Explorations*, Vol 2(2), pp. 1-58.

A Hossain, AM Blackford and K Ward (2011). *Red Team Model Analytical Engine: Prototype Implementation and Requirements Specification.* DSTO General Document, DSTO-GD-0637. Joint Operations Division, Edinburgh, Adelaide.

INCOSE (2007). *Systems Engineering Vision 2020*. INCOSE-TP-2004-004-02, version 2.03, September 2007. Released by Technical Operations, International Council on Systems Engineering.

P Jackson (1998). *Introduction to Expert Systems*, 3rd Edition. Addison Wesley, Boston, MA.

T Mitchell (1997). *Machine Learning*. McGraw Hill, Maidenhead, UK.

North Atlantic Treaty Organisation (2011). *NATO Joint Military Symbology: APP-6 (C)*. NATO Standardization Agency.

S Rajesh, PJ Davies and NJ Curtis (2010). *A Blue Force Activity Analysis Model to Identify S&T Priorities to Enhance Counter IED Capability.* DSTO Technical Report, DSTO-TR-2447. Joint Operations Division, Edinburgh, Adelaide.

LP Kaelbling, ML Littman and AW Moore (1996). Reinforcement Learning: A Survey. *Journal of Artificial Intelligence Research*. Vol 4, pp. 237-285.

L Rayner (2010). *Operations Funding and Enhanced Force Protection in Afghanistan*. Parliamentary Services Briefing Statement to the 2010-11 Budget Review. http://www.aph.gov.au/About_Parliament/Parliamentary_Departments/Parliamentary_Library/pubs/rp/BudgetReview201011/DefenceOpsFunding, Accessed January 1, 2012.

G Robinson(2011). TTSC Brief to JTLS IUC on 25 October 2011. Joint Warfare, Doctrine and Training Centre. Australian Defence Force, Williamtown, Australia.

L Rokach and O Maimon (2005). Top-down Induction of Decision Trees Classifiers: A Survey. *IEEE Transactions on Systems, Man, and Cybernetics. Part C: Applications and Reviews*. Vol 35(4), pp. 476-487.

N Sayers & T McKay (2011). *A Framework for the Design of Red Teaming Exercises*. DSTO Technical Report, DSTO-TR-2621. Joint Operations Division, Edinburgh, Adelaide.

MJ Skroch (2009). Modeling and Simulation of Red Teaming, *Red Team Journal*.

US Department of Defence (2007a). *Architecture Framework, Volume I: Definitions and Guidelines,* Version 1.5, 23 April 2007.

US Department of Defence (2007b). *Common Warfighting Symbology: MIL-STD-2525C*. International Organization for Standardization.

G Wang (2008). A Survey on Training Algorithms for Support Vector Machine Classifiers. *Proceedings of the Fourth International Conference on Networked Computing and Advanced Information Management*. Vol 1, pp. 123-128.

S Wheeler (2006). An Analysis of Combined Arms Teaming for the Australian Defence Force. *Journal of the Operational Research Society*. Vol. 57, pp. 1279-1288.

S Wheeler (2010a). *A Strategy to Task Analysis for JP 154 Phase 2: Joint Counter Improvised Explosive Device*. Defence Science and Technology Technical Note, DSTO-TN-0979.

S Wheeler (2010b). *Use of Scenarios in Support to Operational Concept Document Development: Application to JP 154*. Defence Science and Technology General Document, DSTO-GD-0622. Joint Operations Division, Fairbairn, Australia.

S Wheeler (2011). *Route Clearance for Improvised Explosive Devices*. DSTO Technical Note, DSTO-TN-1018. Joint Operations Division, Fairbairn, Canberra.

S Wheeler (2012). *Moving Forward with Computational Red Teaming*. DSTO Technical Report DSTO-TN-1104. Joint Operations Division, Fairbairn, Canberra.

X Wu, V Kumar, JR Quinlan, J Ghosh, Q Yang, H Motoda, GJ McLachlan, A Ng, B Liu, PS Yu, ZH Zhou, M Steinbach, DJ Hand, D Steinberg. Top 10 Algorithms in Data Mining. *Knowledge and Information Systems*. Vol 14, pp. 1-37.

GP Zhang (2000). Neural Networks for Classification: A Survey. *IEEE Transactions on Systems, Man, and Cybernetics. Part C: Applications and Reviews*. Vol 30(4), pp. 451-462.

Appendix A: Design Brief

JOD runs a CRT Program under Corporate Enabling Research, aiming to future-proof defence against threats posed by Improvised Explosive Devices and those who employ them.

In accomplishing this goal, the Program will deliver a CRT capability to assess technologies and systems, concepts, and force structures. This capability will be realised in the development of a suite of software tools, through means of conceptual design and software prototyping.

Through application of these software prototypes to Red Teaming activities, the Program will enhance the quality of decision making in the ADF. It will inform robust, reasoned and rigorous decision making, in support of current Australian military operations, and the development of capability for the conduct of those operations in the future.

Appendix B: Operational Needs

B.1 Operational Deployment

Expansion of Need

The intended clients for the software prototype are quite broad, being the ADF. However, within the ADF there are three principal stakeholders which might immediately benefit from the software. These include: the Capability Development Group (CDG) in developing projects such as JP154 *Joint Counter IED*; the Counter IED Task Force who are ongoing clients of JOD; and targeted parties within the Vice Chief Defence Force (VCDF) group who assess joint capability. These clients operate on the Defence Restricted Network and Defence Secret Network. JOD operates on its own DSTO Networks. Eventually, both of these options must be supported. However, the prototype will be developed in-house to demonstrate proof-of-concept.

Derived Root Statements[16]

- Client Identity;
- Standard Operating Environment;
- Physical Location of Installation;
- Time of Deployment.

Operational Needs

1.1 The Prototype shall be interoperable with the DSTO Network.
Clarifying statements: Compliant with DSTO Standard Operating Environment.
Built, installed and executed under those available resources.

1.2 The Prototype shall be installed on DSTO encrypted Laptop.
Clarifying statements: Approved by Science Corporate Information Services.
For portable (standalone, non-networked) deployment.

Additional Notes

There is currently no guidance regarding delivery schedule because the schedule depends on available funding and staff resources. No Operational Needs are developed. However, it would be indicative to place a 4-6 year timeframe on the development of the prototype system. Any longer and the original client needs may have changed.

[16] Not all root statements have corresponding needs. For example, it is not necessary to develop a need to identify the client, unless that client specifically impacts on the design. In which case, the need would still be more likely to describe the requirement itself.

B.2 Mission Profile

Expansion of Need

The final software product will eventually be used by defence clients and other stakeholders, independent of DSTO. However, the Prototype system will be employed in a customer-assisted environment. This assumes that DSTO analysts will either run the Prototype on behalf of the client during live activities and experimental campaigns, or employ the Prototype off-line (either before or after the client activity).

Derived Root Statements

- Usability Considerations.
- Fitness-For-Purpose.
- Assisted and Stand-Alone Use.
- Provides a Value-added Product.

Operational Needs

2.1 The Prototype shall present a Graphical User Interface.
Clarifying statements: Capable of visualisation during live client activities.
Operated by trained DSTO analysts.

1.2 The Prototype shall assess technologies and systems for CIED operations.
1.3 The Prototype shall assess concepts and TTPs for CIED operations.
1.4 The Prototype shall assess force structures for CIED operations.
Clarifying statements: Capable of supporting live client activities in near real time.
Operated by trained DSTO analysts.

Additional Notes

Familiarity with the software and a minimum level of training is assumed.

B.3 Performance Parameters

Expansion of Need

The final software product is likely to be deployed through online services across the defence intranet. This will allow a defence user on a remote restricted or secret system to access software on the DSTO network through their internet browser. In this case, the number of concurrent users will guide the development of the network protocol, database administration, and support infrastructure. However, the Prototype will not be developed to support this type of access. It is most likely to be deployed in a DSTO facility such as the JDSC. It will still need to support multiple instances of the software running on one machine and the ability to dock screens across multiple monitors.

Derived Root Statements

- Remote access.
- Flexible display and custom docking of screens.

Operational Needs

3.1 The Prototype shall support multiple concurrent instances of the software.
Clarifying statements: Running on the same server.

3.2 The Prototype shall follow a modular design.
Clarifying statements: Supporting screens docking over multiple monitors.
To be viewed via an overhead projector.

Additional Notes

A modular design is also desirable in that it minimises the amount of concurrent resources required to develop the software. The Prototype can be developed in stages, with minimum commitment from JOD. Further, each component can be rigorously tested as they are developed. For use in many activities, the complete set of software components may not be required. In this case, there are additional efficiencies in deploying a minimalist design process.

B.4 Utilisation

Expansion of Need

The final product should be available to defence clients 75 hours a fortnight, during standard business hours. This is indicative of a mature service provider and is likely to include support such as a helpdesk hotline and training courses. The Prototype need only be available during pre-planned activities and will not provide any additional support services.

Derived Root Statements

- Availability.
- Helpdesk Services.
- Training Services.

Operational Needs

4.1 The Prototype shall be available for use with two weeks prior notification.
Clarifying statements: Corresponding to the average warning-time provided to JOD prior to running a client activity.

Additional Notes

Operational Need 4.1 is likely to be achieved with the release of two candidate Prototypes. The first will be the stable build and will be deployed for customer use. The second will be the developmental build and will be used for experimental purposes only.

B.5 Effectiveness

There are real effectiveness considerations for the use of the Prototype. The development process itself does not have to show an economic benefit; in that, the software will not be sold for profit. However, there are concerns regarding the provision of the raw and processed data which must be gathered for the software to be instantiated. Information repositories in the final product must be fully maintained by the client. The Prototype will be instantiated by JOD.

Derived Root Statements

- Instantiation.
- Information Repositories.
- Maintenance and concurrency.

Operational Needs

4.1 The Prototype shall following industry standards in database design.
Clarifying statements: Initially to be designed, implemented and populated by JOD.

Additional Notes

Storage of information will not be a problem. Servers in the JDSC are more than adequate for this task. However, there may need to be a separate (minimalist) instantiation for stand-alone deployment.

Appendix C: Functional Analysis

In this appendix, enhanced functional flow block modelling is applied to conceptually describe one possible design for the model. An abbreviated systems engineering process has been followed to functionally decompose the model. Standards across the industry vary; we have adopted the following symbology.

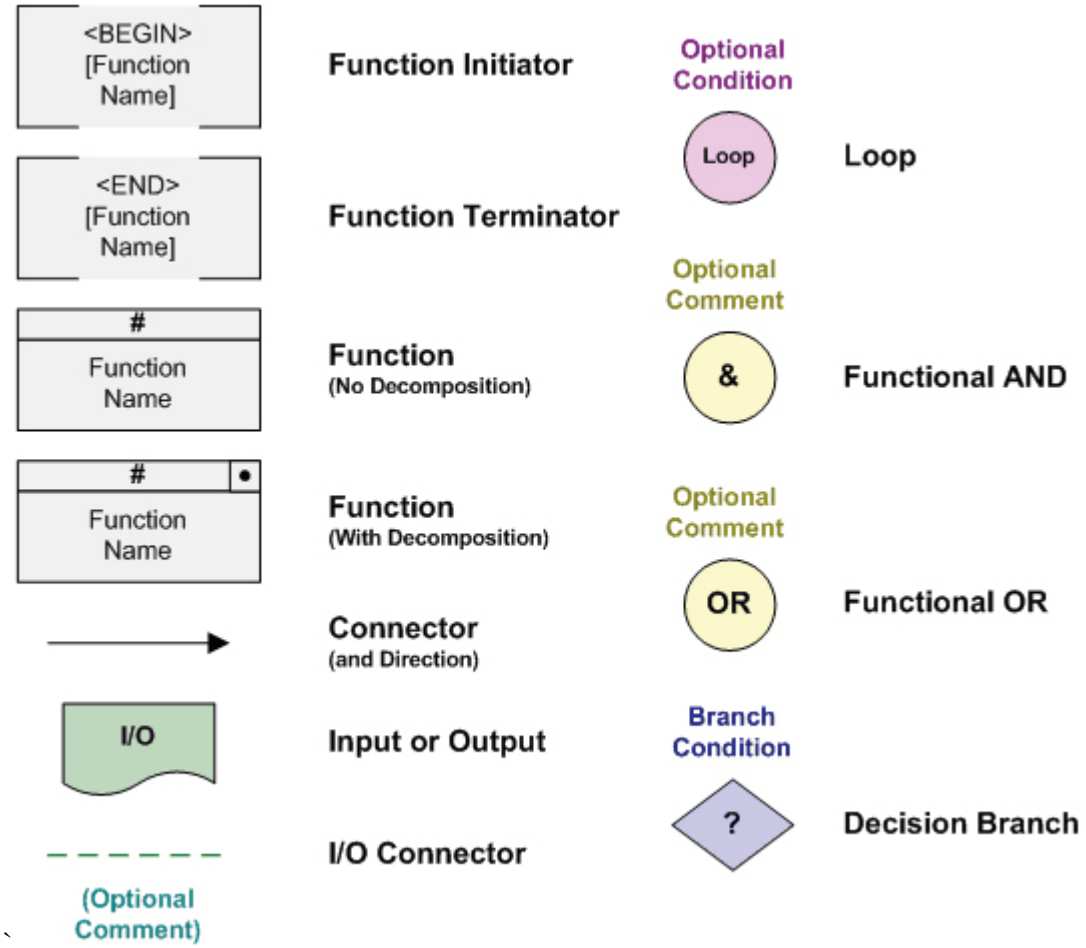

Figure 8: Legend - Enhanced functional flow block diagram

C.1 Top Level Design

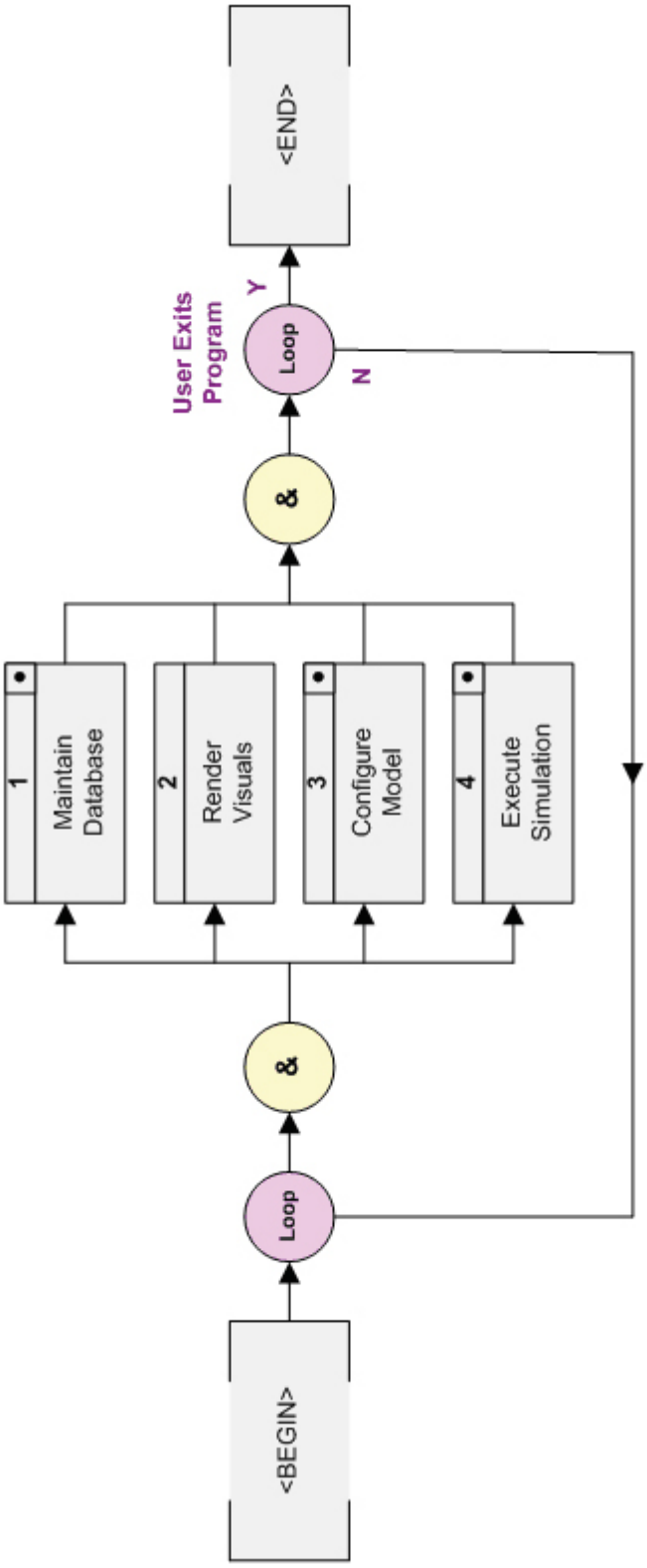

Figure 9: Top level design

The top level design for the model is decomposed into four key functions.

1. *Maintain Database*

   The prototype should be capable of loading and saving models. The format of the database is not specified in this document and may potentially be implemented using rudimentary techniques; such as structured text or comma separated values files; or a modern database proper. It is envisaged that a simple implementation may be adopted initially, allowing for development of a formal database once proof-of-concept for the prototype system has been demonstrated.

   The database should be capable of storing metadata (parameters specific to the software execution of the program itself, e.g. display size). The purpose of storing this data is to maintain user profiles. Eventually, security considerations will also need to be addressed when operational data is to be stored. The initial design should be mindful of the implications imposed by this constraint.

   Simulation logs should be captured within the database for the purposes of replaying simulations and conducting analysis offline.

2. *Render Visuals*

   It is desirable that the software support a graphical user interface for model design and database manipulation. It is also desirable that the prototype be capable of rendering a visual display of the state of the model.

   The specific requirements for the prototype system are not yet defined so it is too early yet to say if either of the requirements above is mandatory. It is certainly possible to implement the core system without the convenience of modern visuals. Eventually, however, a mature system will need to address the usability of the software and its accessibility to a wide audience.[17]

   The functions above may not necessarily be implemented by the same component. It may prove to be simpler to implement the three functions (model design, database manipulation, and simulation display) as independent applications. This also offers a development path with the efficiency of a modular design.

3. *Configure Model*

   Configure Model encapsulates all of the functionality in the definition and instantiation of entities, the simulation environment, and the model. The model is defined as the combination of the entities in their environment, the initial state of the model, and the set of input parameters. Metadata may also be stored which impacts on the execution of the model (for example, termination criteria) but this information may be a part of the global configuration of the software.

[17] Those being any individual outside the prototype design team.

The end user may load an existing model or save currently open models. There may not be a limit on the number of models which can be opened at any time. The design should cater for multiple instances of open models and the ability to clone parts of one model (for example, entities) for insertion into another.

4. *Execute Simulation*

Execution of the model is performed in this function. The state of the model is updated and logged in the database for record.

Given the objective of the prototype; that being to develop a simple model for capability assessment; this first model is likely to be scripted. This means that many of the variables (for the behaviours of the entities over time) will be fixed with only a few free parameters. Execution of such a model could then simulate a route clearance operation, for example, where the process is identified upfront (according to doctrine) and the level of capability is varied. This model could be used to critique red or blue plans during an activity.

The prototype will eventually be an interactive model. End users should be able to pause, rewind or fast-forward simulations after execution.

C.2 Serial 1: Maintain Database

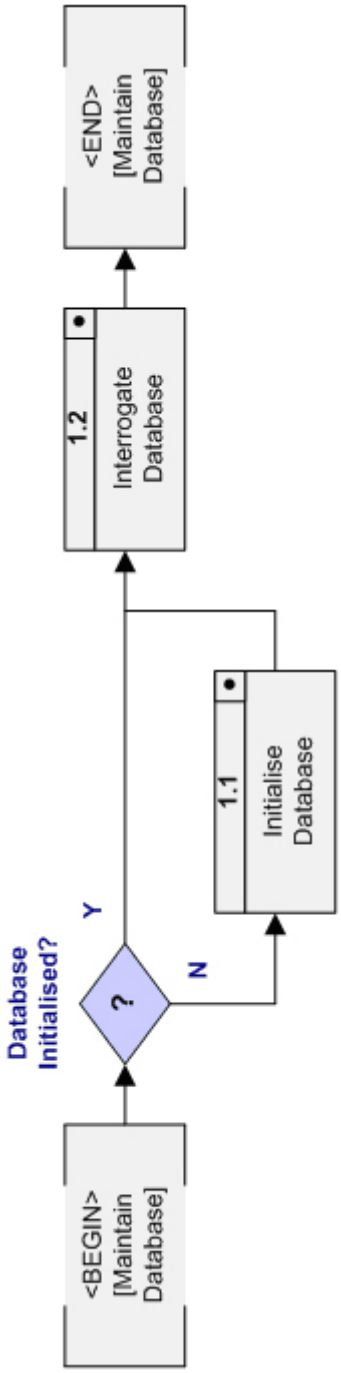

Figure 10: Serial 1: Maintain database

Serial 1: Maintain Database is decomposed into two main functions.

1. *Initialise Database*

   Initialise Database allocates an empty storage space, if one has not already been defined. As in the explanation of the Maintain Database function, this storage space is generic in that it does not necessarily have to be implemented as a formal database.

   The data will be structured irrespective of the means of implementation. This structure is assumed to be pre-defined by a schema.

   This function also establishes a datalink to the database for read and write access. The datalink provides a mechanism by which other components of the prototype can access the database.

2. *Interrogate Database*

   The interrogation of the database utilises the datalink to access and commit data.

   No distinction is made as to whether this function is conducted as a result of a direct end-user action or as a result of the execution of a model. The database should maintain its own consistency and will conduct a number of automatic actions. For example: maintaining a log of simulation output; updating user configurations when changes are made; and storing user access data for security purposes. These items are explained in greater detail under Serial 4: *Execute Simulation*.

The decomposition of Serials 1.1 and 1.2 are provided in the two figures below.

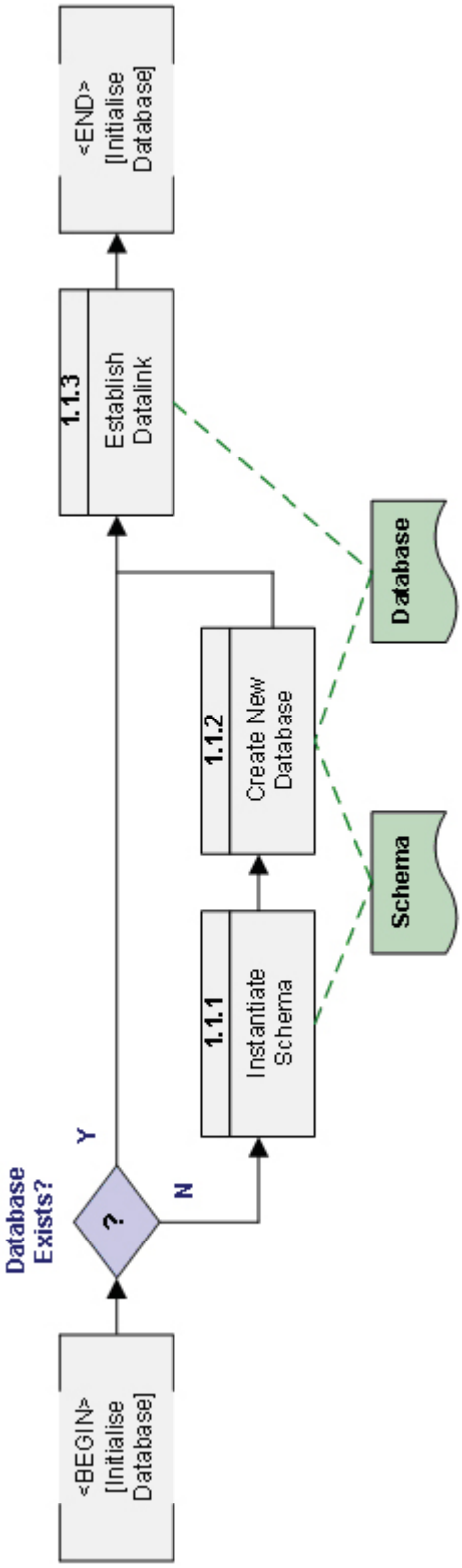

Figure 11: Serial 1.1: Initialise database

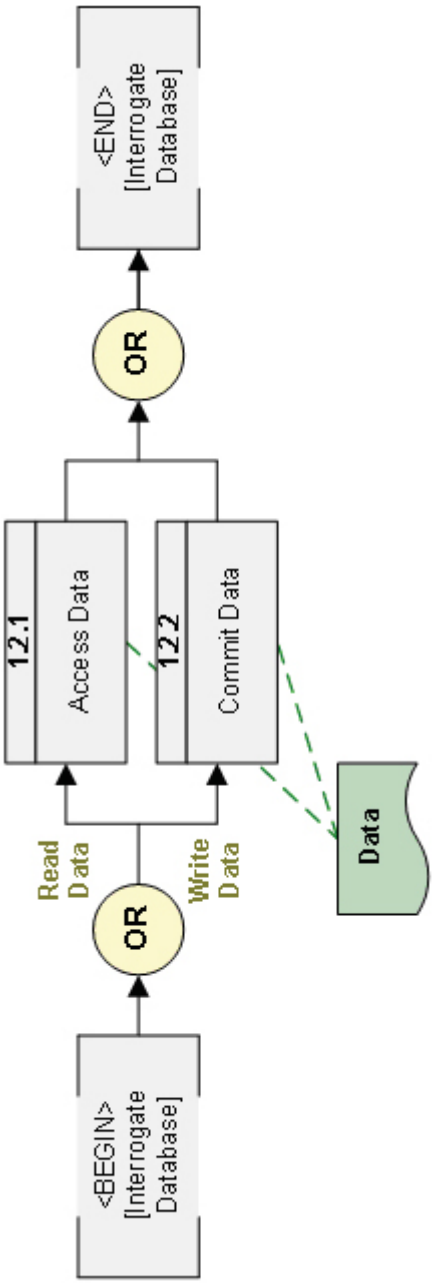

Figure 12: Serial 1.2 Interrogate database

C.3 Serial 3: Configure Model

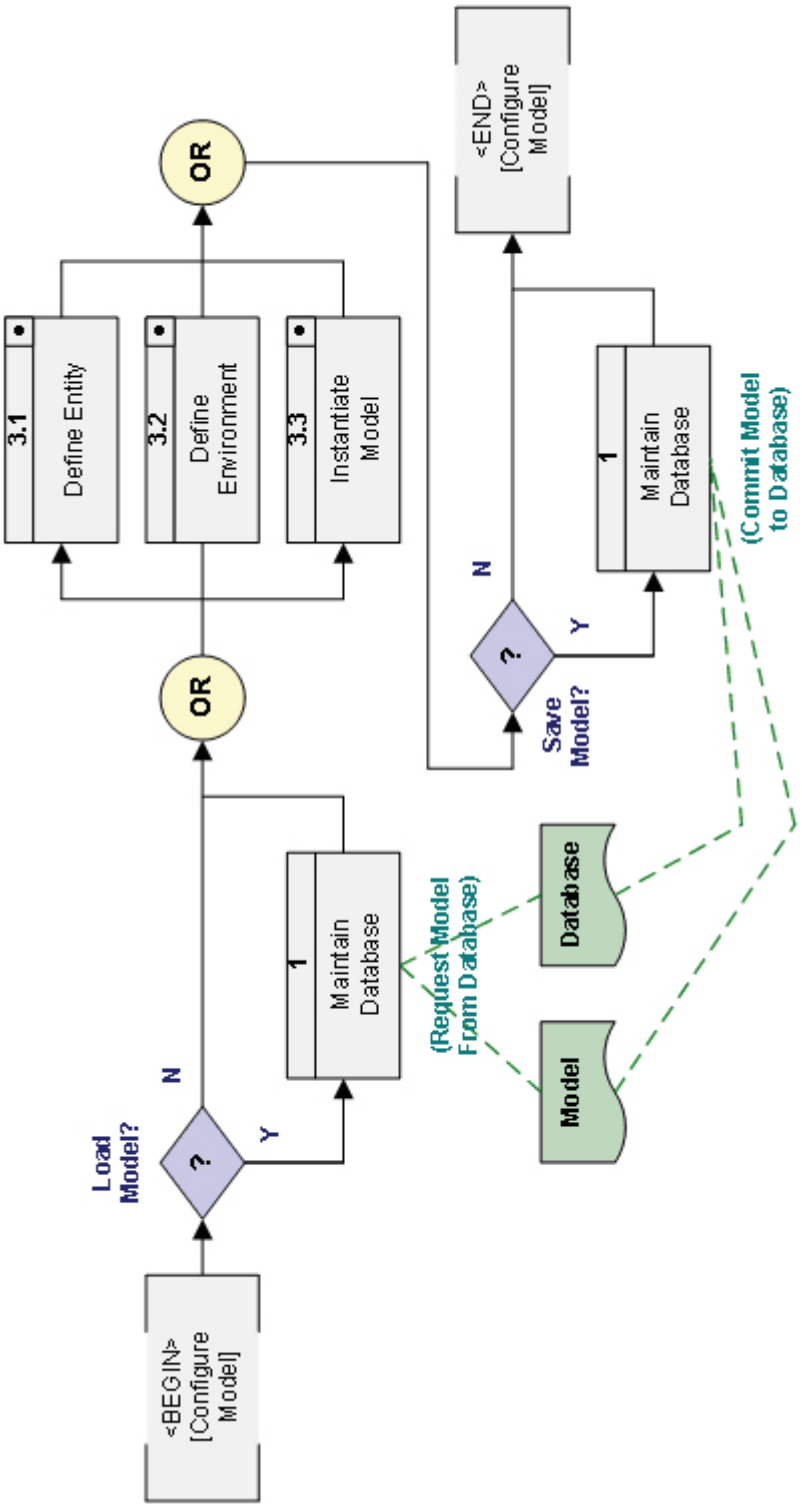

Figure 13: Serial 3: Configure model

Serial 3: Configure Model is decomposed into several main functions.

1. *Define Entity*

   Entities are described in terms of a collection of properties and behaviours. Entities are then the set of all interacting components of the model which require specification.

   For example, a Main Battle Tank might be defined in terms of its mobility and lethality. Behavioural rules for the tank are also be defined here. These rules describe how those attributes are employed. In this example, those rules would encode movement and engagement behaviours.

2. *Define environment*

   The environment is described as the set of permissible states for all entities. This includes aspects of the operating environment, such as terrain, and also restrictions over the environment, such as boundaries. These are the rules applied within the model which regulate the possible values and combinations of values.

   Definition of the environment also includes an Order of Battle (ORBAT). The ORBAT identifies the number of different types of entities in a force, where each entity itself has been defined above.

   Metadata, attached to the model, may also be recorded. Metadata is not specifically related to the entities but to the execution of the simulation. This might include termination criteria, data-logging requirements, model name and creator, date and time, and other custom parameters.

3. *Instantiate Model*

   The actual initial state of the model is defined in this function. This state will comply with the rules defined above for valid combinations of parameters.

   For example, the initial state might define the tactical disposition of all forces.

Additionally, the function *Maintain Database* also appears (defined as Serial 1). It is assumed that end-users will save and load models in the database. This functionality supports the development of models over more than one session of use. It is assumed that end-users will wish to store a range of models and customise new ones from existing templates.

The decomposition of Serials 3.1 to 3.3 is provided in the two figures below.

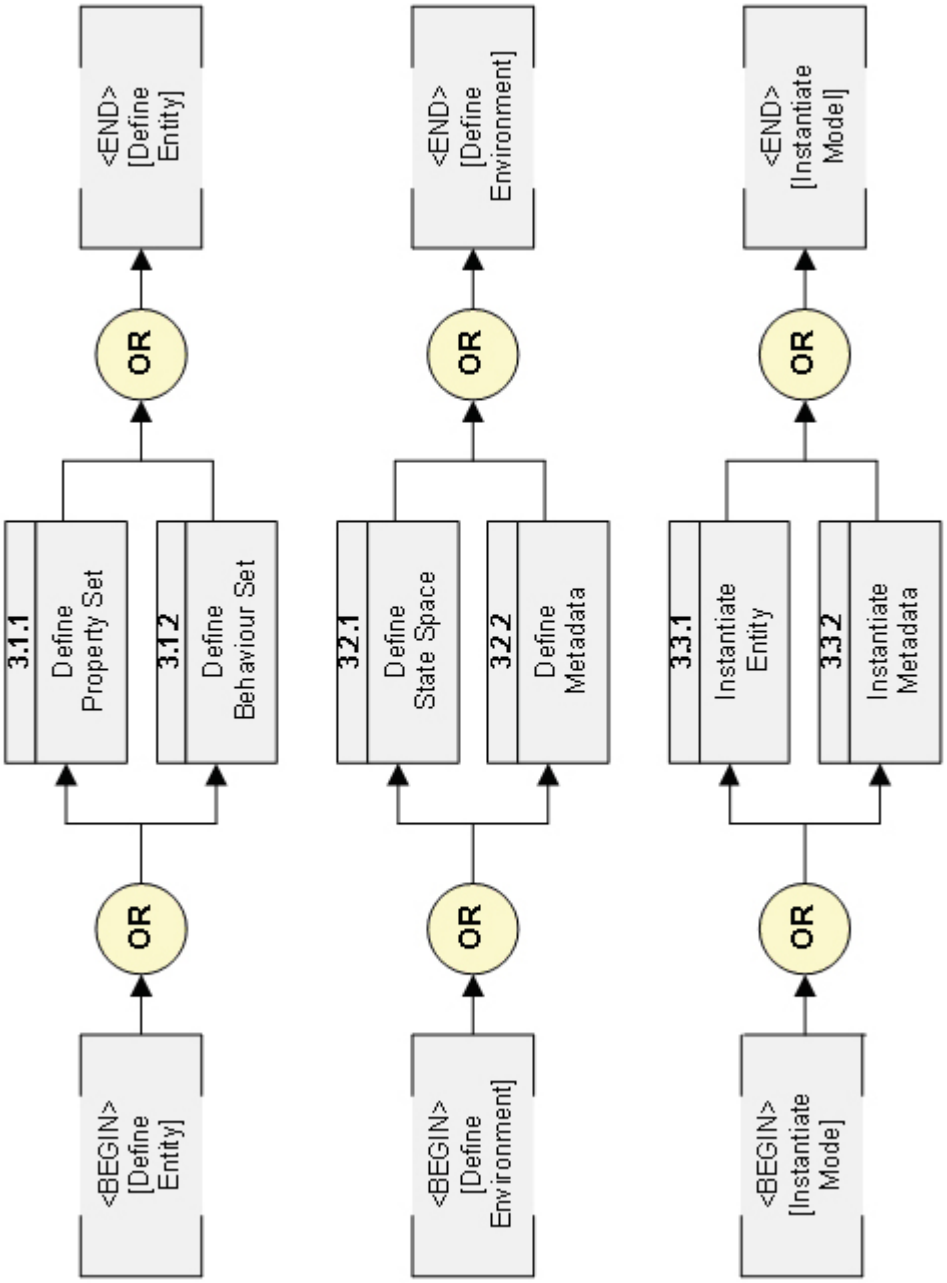

Figure 14: Serials 3.1.1, 3.1.2, 3.1.3: Define entity; define environment; instantiate model

C.4 Serial 4: Execute Model

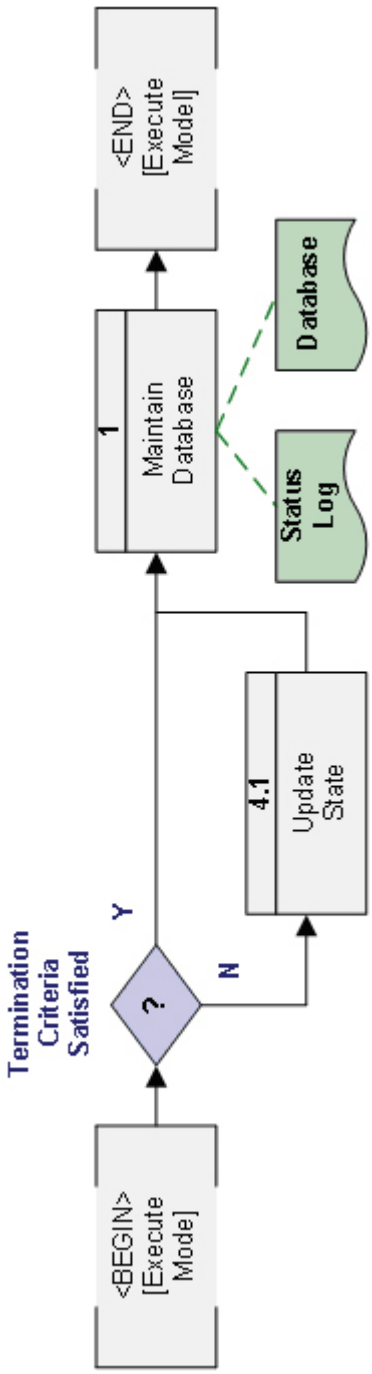

Figure 15: Serial 4: Execute model

Serial 4: Execute Simulation is decomposed into one primary function.

1. *Update State*

   This function assumes that the prototype is to be a discrete event simulation. There are two possibilities. First, the simulation is discrete time iterated. Second, the simulation is event driven.

   Both options come associated with performance and deployment considerations. The event driven simulation is generally more efficient and easier to deploy in a parallel processing environment, cloud computing suite or super computer hive. However, depending on the specific requirements for the application, event driven simulation may not be appropriate.

   It is also possible that the prototype will not be a discrete event simulation. A geographic profiling application is a reasonable example. The state of the simulation in time may then be degenerate when only one point in time is modelled. For example, calculating a single geographic 'heat-map' in a given area against calculating the change in heat-map for that area over a time interval. The first case is degenerate (modelling a time interval of a single step).

Additionally, the function *Maintain Database* also appears (defined as Serial 1). Here, the function is used to generate a simulation log. The log can be used for replay, analysis, and debugging.

Appendix D: Component Allocation

This appendix presents the allocation of functional blocks to their respective components. Standards across the industry vary; we have adopted the following symbology.

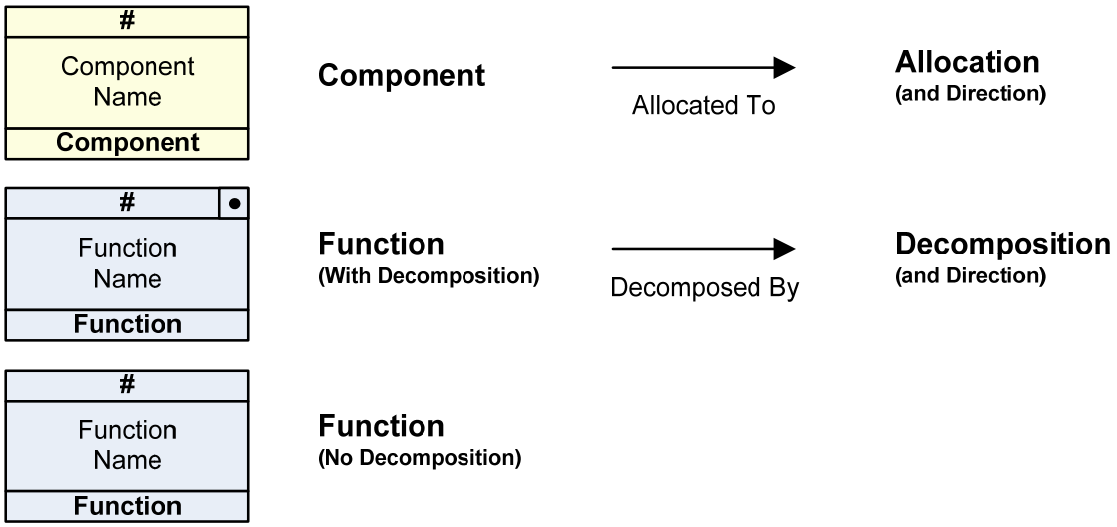

Figure 16: Legend: Functional component allocation model

D.1 Serial 1: Maintain Database

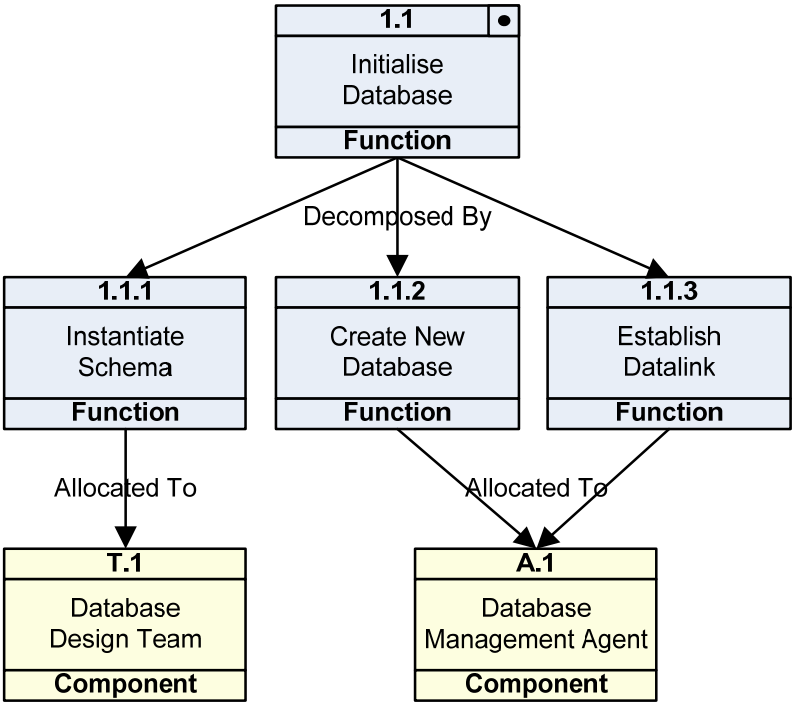

Figure 17: Component allocation for Serial 1.1: Initialise Database

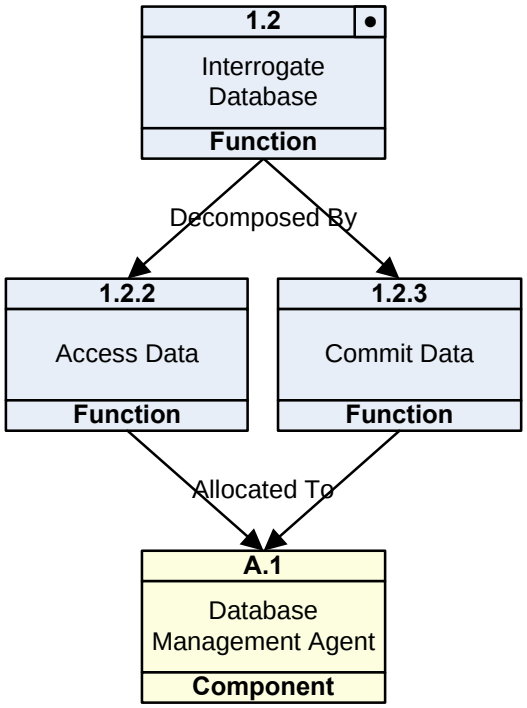

Figure 18: Component allocation for Serial 1.2: Initialise Database

Two components are introduced in implementing the function Maintain Database.

1. *T.1: Database Design Team*

   The database design team is constructed of one or more people. It is assumed that these will be DSTO staff, although contracted services are also a possible alternative.

   This team is responsible for establishing and evolving the database. They will define the initial schemata for the database and recommend the initial implementation. To this end, the team shall be sufficiently experienced to make these recommendations.

   Amongst options to be considered, the database may be implemented as a simple text file. This is also intended to be a temporary design. Other options include commercial suites, Microsoft Access being an obvious candidate as it is available on the defence standard operating environment.

   Specialised database formats might also be considered. Should the body of the prototype be developed in say Matlab, then the Matlab load/save workspace routines could easily be customised to fulfil the database requirements.

2. *A.1: Database Management Agent*

   The database management agent is distinct from the design team. Whatever format the database takes, there shall be automated functions to access and store data. This agent would facilitate this access in real time to support operations conducted during execution of simulations.

   In some sense, what this agent provides is a wrapper and advanced programming interface, to the database functionality for integration with the software. These functions include creating a new instance of a database, providing a link or handler to the software, facilitating data access and committing data.

D.2 Serial 2: Render Visuals

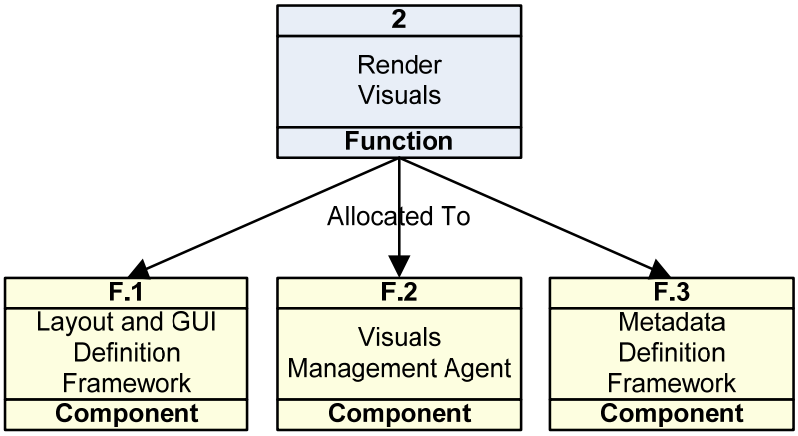

Figure 19: Component allocation for Serial 2: Render Visuals

Three components are introduced in implementing the function Render Visuals.

1. *F.1: Layout and GUI Definition Framework*

   The user interface may not be implemented in the prototype. However, when completed it should support customisable user-defined layouts.

   This function is complex in that layouts might be configured based on whether the user is simulating the model, developing a model, or analysing results. Different configurations will be stored as metadata against the user profile.

   It is also possible that the user interface will be implemented as a federation of separate windows, one for each configuration. Alternatively, a tabbed environment could be used within a single window.

   In the prototype, separate windows with fixed configuration are appealing. This permits separate (semi-independent) development of each window. This is useful because the resources (staff and capital) are likely to be limited in the implementation of the prototype. Hence, development efforts can be more easily managed with a phased budget over several accounting periods.

2. *F.2: Visuals Management Agent*

   The visuals management agent is responsible for refreshing and displaying information to the screen. The actual implementation might not display every change in state during the simulation. Depending on the number of events triggered each second, some changes in state might be dropped. The prototype also might not be capable of displaying the simulation in real time. If this is the case, the visuals management agent might only need to display a replay screen (and not a

simulation screen). The difference being that a simulation screen might permit the user to interact with the model in real time while the replay does not.

3. *F.3: Metadata Definition Framework*

   Metadata is any type of information which affects the modelling environment or the execution of the simulation but which is not used as an input to the model itself. This includes records such as user preferences and execution control information such as number of replications of each simulation and termination criteria. This framework provides the interface between the user and the software environment in which metadata is defined.

   It is reasonable to expect that the framework will be arranged according to similarity, perhaps in separate tabs. This implies the metadata relating to the user profile would be presented on the same tab and metadata relating to the model execution on another.

   If such a design is adopted then the framework itself may appear in different virtual windows. This would mean that the tabs are split and may only be accessible in an appropriate context. For example, the metadata according to the execution of the model might only be accessible from the modelling window.

D.3 Serial 3: Configure Model

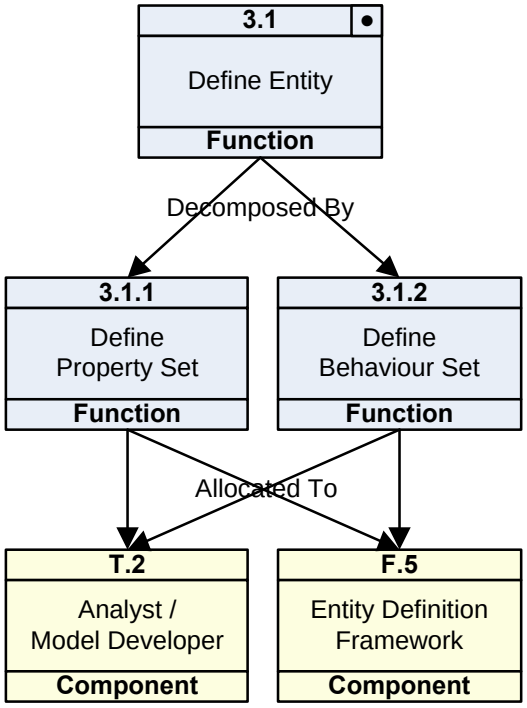

Figure 20: Component allocation for Serial 3.1: Define Entity

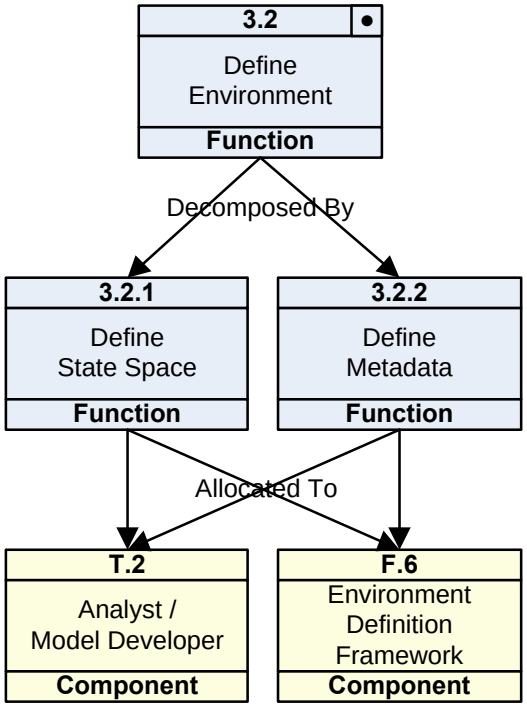

Figure 21: Component allocation for Serial 3.2: Define Environment

Two components are introduced in implementing the function Define Entity.

2. *T.2: Analyst / Model Developer*

   The analyst / model developer will initially be the actual user of the prototype. In most instances they will be responsible for defining the model itself. There is also a minor differentiation to be made here. The model developer will define and establish the model. The analyst might use the prototype for the purposes of generating results. In many instances, these will be the same person.

   This might not be the case when the software is deployed to support a Red Teaming activity. In this instance, trained personnel might run the software, having prepared the model in situ. Here, the individual or team running the prototype would only need to be trained in its use.

3. *F.6: Entity Definition Framework*

   The entity definition framework describes the basic mechanism by which the entity properties and behaviours are defined. This should be implemented by a software engineer or coder. The specific form implementation is somewhat open but there are two principle options. First, a scripting language could be interpreted by the software at runtime (or perhaps compiled) in a manner similar to programming MATLAB or NETLOGO. Second, a fixed number of options could be pre-encoded (hard coded) into the prototype in a manner similar to MANA.

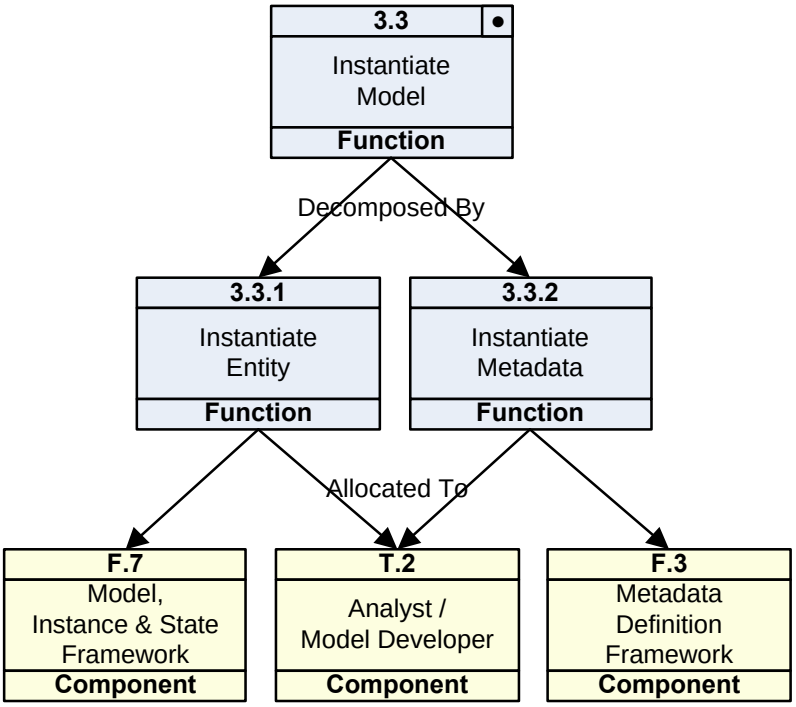

Figure 22: Component allocation for Serial 3.3: Instantiate Model

Three components are introduced in implementing the function Instantiate Model.

1. *T.2: Analyst / Model Developer*

   The analyst / model developer component has already been described under Figure 21.

2. *F.3: Metadata Definition Framework*

   The analyst / model developer component has already been described under Figure 21.

3. *F.7: Model, Instance & State Framework*

   The model, instance & state framework provides the technical means by which the analyst or model developer defines entities. Specifically, the properties and behaviours associated with each entity in the model are defined here.

   This framework is also used to define the collection of entities and their state. This functionality is described in Figure 23 below.

D.4 Serial 4: Maintain Database

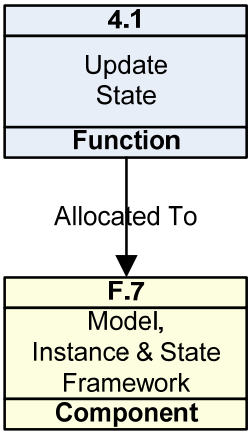

Figure 23: Component allocation for Serial 4.1: Update State

1. *F.7: Model, Instance & State Framework*

   Component F.7 has been previously discussed in the previous sub-section. In the context of Serial 4.1 Update State, it specifically records the properties of the model, inclusive of all entities, the environment and other variables. These parameters vary in time. However, this component is not responsible for storing the time series of these parameters because that is performed by the database management agent.

Appendix E: Subject Matter Contribution

List of parties who have contributed to the outcomes of the report

Specialist Consultants

Name	Role	Organisation
Dawn Hayter	Managing Director	IGOR Human Sciences
John Wiese	Director Defence Systems	Thales Australia Perth
Tess McCarthy	Criminal Intelligence	Australian Federal Police

DSTO Subject Matter Experts

Name	Role	Organisation
Andrew Gill	Lead Geographic Profiling	JOD Edinburgh
Gary Bullass	Lead Discovery Team	JOD (JDSC) Fairbairn
Jon Rigter	Lead Systems Studies	JOD (JDSC) Fairbairn
Martin Wong	Software Engineer	LOD Edinburgh
Paul Gaertner	Head Emerging Technology	JOD Edinburgh
Piers Duncan	Lead Red Teaming	MOD Eveleigh

Internal JOD CRT Program

Name	Role	Organisation
Coen van Antwerpen	Lead Corporate CRT	JOD Edinburgh
Greg Newbold	a/Lead JP154-CRT	JOD Edinburgh
Jennie Clothier	Chief JOD	JOD Canberra
Nathan Sayers	Development Lead Models	JOD Edinburgh
Paul Whitbread	Lead Divisional CRT	JOD Fairbairn
Phil Gowlett	Science Lead CRT [prior]	JOD Edinburgh
Phil James	Research Leader COPS	JOD Edinburgh
Terry Moon	Lead CRT Review	JOD Edinburgh
Tim McKay	Research Leader FOPS	JOD Canberra

DEFENCE SCIENCE AND TECHNOLOGY ORGANISATION
DOCUMENT CONTROL DATA

1. PRIVACY MARKING/CAVEAT (OF DOCUMENT)

2. TITLE

Future Proofing Computational Red Teaming

3. SECURITY CLASSIFICATION (FOR UNCLASSIFIED REPORTS THAT ARE LIMITED RELEASE USE (L) NEXT TO DOCUMENT CLASSIFICATION)

Document	(U)
Title	(U)
Abstract	(U)

4. AUTHOR(S)

Scott Wheeler

5. CORPORATE AUTHOR

DSTO Defence Science and Technology Organisation
Fairbairn Business Park Department of Defence
Canberra ACT 2600 Australia

6a. DSTO NUMBER	6b. AR NUMBER	6c. TYPE OF REPORT	7. DOCUMENT DATE
DSTO-TN-1109	AR-015-365	Technical Note	August 2012

8. FILE NUMBER	9. TASK NUMBER	10. TASK SPONSOR	11. NO. OF PAGES	12. NO. OF REFERENCES
2012/1051949/1	CDF 07/331	COMD CIED TF	65	28

13. DSTO Publications Repository

http://dspace.dsto.defence.gov.au/dspace/

14. RELEASE AUTHORITY

Chief, Joint Operations Division

15. SECONDARY RELEASE STATEMENT OF THIS DOCUMENT

Approved for public release

OVERSEAS ENQUIRIES OUTSIDE STATED LIMITATIONS SHOULD BE REFERRED THROUGH DOCUMENT EXCHANGE, PO BOX 1500, EDINBURGH, SA 5111

16. DELIBERATE ANNOUNCEMENT

No Limitations

17. CITATION IN OTHER DOCUMENTS Yes

18. DSTO RESEARCH LIBRARY THESAURUS http://web-vic.dsto.defence.gov.au/workareas/library/resources/dsto_thesaurus.shtml

Computational Analysis, Decision Support Systems, Red Teaming, Automated Reasoning

19. ABSTRACT

Computational Red Teaming describes the application of new and innovative analytic techniques, tools and methodologies in support of Red Teaming Activities. This report documents the formal design of such a tool, to inform the future direction of the DSTO research program. Requirements for an executable prototype are presented to construct the initial design, developed through the application of Systems Engineering practices. Recommendations to further develop the prototype are provided, with illustrative examples. When implemented, the prototype will deliver a Computational Red Teaming capability to assess technologies and systems, concepts and force structures; with the potential to inform better decisions in future Australian capability development.